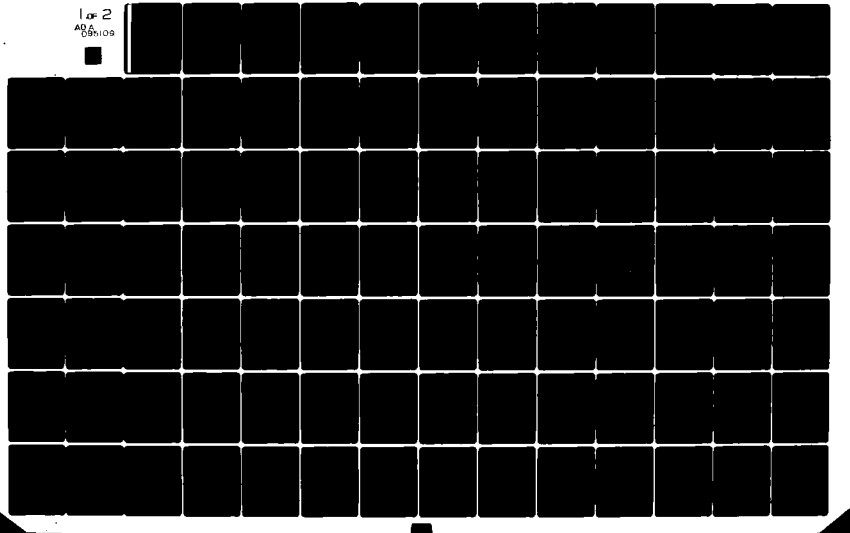
AD-A095 109 ENVIRONMENTAL RESEARCH INST OF MICHIGAN ANN ARBOR RA--ETC F/G 20/6
DESIGN CONCEPTS FOR AN OPTICAL NUMERICAL COMPUTER BASED ON THE --ETC(U)
JUN 79 A M TAI, I CINDRICH, J R FIENUP DAS660-78-C-1035
UNCLASSIFIED ERIM-136800-14-F NL



136800-14-F

Final Report

DESIGN CONCEPTS FOR AN OPTICAL NUMERICAL COMPUTER BASED ON THE RESIDUE NUMBER SYSTEM

ANTHONY M. TAI, IVAN CINDRICH,
JAMES R. FIENUP, CARL C. ALEKSOFF
Radar and Optics Division

JUNE 1979

Ballistic Missile Defense Systems Command
Advance Technology Center
P.O. Box 1500
Huntsville, Alabama 35807

ENVIRONMENTAL
RESEARCH INSTITUTE OF MICHIGAN
BOX 8618 • ANN ARBOR • MICHIGAN 48107

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO. AD-A093409	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Design Concepts for an Optical Numerical Computer Based on the Residue Number System.	5. TYPE OF REPORT & PERIOD COVERED Final Report. Aug. 1978 – May 1979
	6. PERFORMING ORG. REPORT NUMBER 14 ERIM-136800-14-F
7. AUTHOR(s) Anthony M. Tai, Ivan Cindrich, James R. Fienup, and Carl C. Aleksoff	8. CONTRACT OR GRANT NUMBER(s) DASG60-78-C-1035
9. PERFORMING ORGANIZATION NAME AND ADDRESS ERIM P.O. 8618 Ann Arbor, Michigan 48107	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 6.33.04.A
11. CONTROLLING OFFICE NAME AND ADDRESS Ballistic Missile Defense Systems Command Advance Technology Center P.O. Box 1500, Huntsville, Alabama 35807	12. REPORT DATE June 1979
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME AND ADDRESS *(if different from Controlling Office)*	15. SECURITY CLASS *(of this report)* UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

Ivan Cindrich was the Principal Investigator at ERIM for this project and Joe McKay at BMDATC was the Army Contract Monitor.

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Numerical Optical Processor
Optical Residue Computer
Integrated Optics
Electro-Optics

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

In this report, the residue number system is reviewed and various implementation concepts for a residue based numerical optical processor are presented. A detailed design for an optical processor based on the mapping approach is given. A preliminary estimate on the performance levels are provided together with recommendations on immediate and future developmental needs and plans.

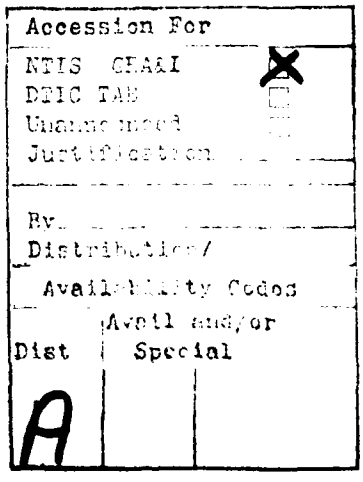
DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

PREFACE

The work reported here was performed at the Electro-Optics Department of the Radar and Optics Division of Environmental Research Institute of Michigan (ERIM). This work was sponsored by the Ballistic Missile Defense Systems Command, Advanced Systems Center.

This final report covers work performed between August 1978 and May 1979. The Contract Monitor is Joe McKay, BMDATC-P, P.O. Box 1500, Huntsville, Alabama 35807. The Principal Investigator is Ivan Cindrich. The major contributors are Anthony M. Tai, James R. Fienup, and Carl C. Aleksoff.



CONTENTS

CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

LIST OF TABLES

1
INTRODUCTION

Optical methods have been successfully applied in a number of data processing applications such as correlation and Fourier transformation operations. Typically, these optical processing techniques are analog in nature and they offer very high processing speed within the optical channel by operating in parallel. For example, the equivalent of 10^6 data samples can readily be Fourier transformed in parallel in a few nanoseconds once the data are entered into the optical processing channel. However, compared to digital devices, the analog optical techniques are more limited in accuracy, flexibility, and programmability. The prospect of combining the parallism and speed of an optical processor with the accuracy and flexibility of a digital machine is a highly attractive concept and may well serve as the design goal of modern computer engineers.

In the design of digital electronic devices considerable effort is directed to increasing processing speed. Higher degrees of parallism are achieved through pipelining and the use of LSI and VLSI technologies. New electronic switching devices are being developed to push the propagation speed closer to that of the speed of light. An alternate approach toward the same goal is to produce a numerical optical processor with the accuracy and flexibility of an electronic digital machine. The possibility of developing such a numerical optical processor is the subject of this study.

A basic tenant for the numerical optical processor considered here is that data are handled in a quantized and encoded form. The encoding would be dependent on the underlying number system. There are several number systems that might be used for a numerical optical processor. However, our present study is directed to the residue number system. The use of the residue number system allows basic arithmetic to be performed without the need for carry operations. Residue arithmetic is also very inducive to parallel architecture in processor design.

The fundamental properties of the residue number system and related computing algorithms were examined in the study and they are reviewed in Section 2. An overview of various numerical optical processor design approaches using residue arithmetic is given in Section 3. In Section 4, we describe in depth the design of an optical processor utilizing the optical mapping approach. The implementation is realized as programmable spatial maps that are built up into a versatile arithmetic module. These computation modules can be interconnected and programmed to perform a variety of more complex processing operations. Section 5 provides a performance level estimate and comparison for the processor design concept introduced in Section 4. A discussion of the developmental needs for the realization of a numerical optical processor is presented in Section 6. The last section of the report, Section 7, provides a developmental plan for the realization of a numerical optical processor.

The processor designs presented in this report are quite specific as to hardware implementations. The purpose is to provide a more solid perspective on the potential capabilities of a numerical optical processor. However, the design concept can be implemented equally well with hardware other than those chosen in this report. Based on the demonstrated performance of the hardware utilized in our design, we are able to show that a processor throughput rate over 300 MHz can be achieved. The versatility of the system is also demonstrated by applying it to various signal processing problems such as matrix multiplication and discrete Fourier transformation. The same high throughput rate is obtained for these computations with the use of parallel structures and pipelining. We should also emphasize that our design concept reflects the stage of present hardware technologies; the design will evolve with the development of hardware components directed specifically to a numerical optical processor.

2
REVIEW OF RESIDUE ARTIHMETIC

2.1 INTRODUCTION

Nearly all of the number systems we generally encounter are weighted number systems and most of them are fixed radix systems (e.g., decimal and binary systems). The residue number system is not a weighted number system and many of the common properties that we are so familiar with no longer apply. The unique characteristics of the residue number system and residue arithmetic provide some very useful properties together with a few troublesome penalties. The major advantage in performing arithmetic operation in the residue number system is the absence of the carry, thereby allowing the computation to be performed in a single step (1 clock cycle). The time saving is particularly pronounced in multiplication operations since the need for partial product is also eliminated.

Carried with these advantages are some consequences of not being a weighted number system. The magnitude of a number with residue representation is not evident from the values of the residue digits. This adds significantly to the complexity of performing many condition checks such as magnitude comparison, sign check, overflow detection and error detection. The residue number system is an integer system and no fractional value can be represented (at least in a straightforward manner). Thus, the operands and the results of any arithmetic operation must be integers. This is especially troublesome for division operation where the quotients are generally fractional values, even when both the operands are integers. Division cannot be carried out without complex procedures and the quotient must be rounded to the closest integer which is smaller than the exact result. Therefore, to fully utilize the speed potential of residue arithmetic, it is usually applied to problems that can be formulated in such a way such that no division is necessary.

Due to the efficiency with which residue arithmetic performs addition and multiplication, computer engineers have, for years, tried to incorporate residue arithmetic in their computer systems.[2-6] While they were successful, residue arithmetic so far has very limited impact in the field of numerical computing. Partly, it is due to the drawbacks we mentioned earlier regarding the residue number system. In addition, the efforts in the past are concentrated in adopting existing computer hardwares for the implementation of residue arithmetic. To fully utilize the advantages offered by residue arithmetic, special hardware and design concepts which are tailored to the unique features of the resiude number system must be developed. The optical approach to implementation offers techniques which in several respects may be especially well suited to the residue number system.

2.2 RESIDUE NUMBER REPRESENTATION

A residue number system is based on N relatively prime integers $m_1, m_2, \cdots m_N$ called moduli. An integer within this number system is represented by a N-tuple of integers $\{r_1, r_2, \ldots r_N\}$ and r_i is defined by the equation

$$x = km_i + r_i, \quad i = 1, 2, \ldots N$$

where k is an integer and $0 \leq r_i < m_i$. If we let $\left[\frac{x}{m_i}\right]$ represent the integer part of the quotient obtained from the division operation $\frac{x}{m_i}$, the residue of x modulo m_i is defined as

$$r_i = |x|_{m_i} = x - \left[\frac{x}{m_i}\right] m_i$$

For example, for a residue number system based on moduli 2, 3, 5, and 7, an integer number x = 14 can be represented as

$$14 = (r_1, r_2, r_3, r_4) = (0, 2, 4, 0)$$

The residue representation of numbers, however, is not unique.

Let us assume that two integers x and x' have the same residue representation $(r_1, r_2, \ldots r_N)$. Since

$$r_i = x - km_i$$

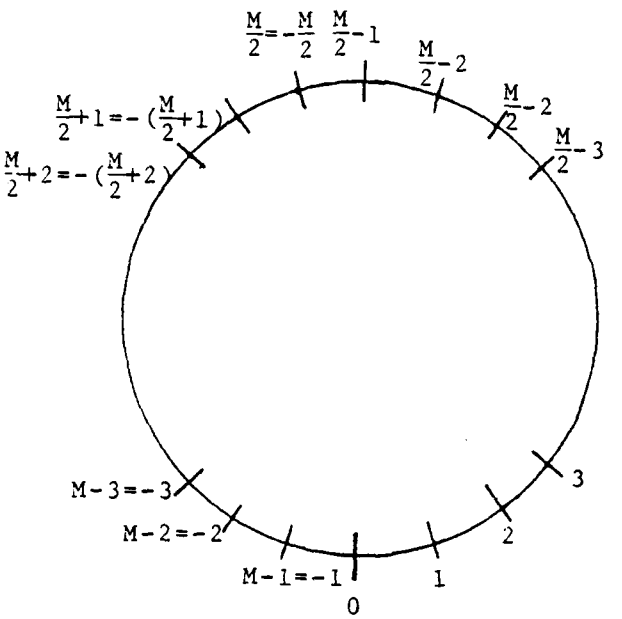
$$= x' - k'm_i$$

we have $x - x' = m_i(k - k')$ for all m_i. $x - x'$ is therefore divisible by all m_i and this would imply that $(x - x')$ is a multiple of M where

$$M = \prod_{i=1}^{N} m_i$$

Thus, the residue representations are the same for integers A, A + M, A + 2M, etc. The residue representation of integers is thus unique only if $(k - 1)M \leq x < kM$. For simplificity, the range $0 \leq x \leq M - 1$ will be used in this report.

Although the residue number system represents only positive numbers explicity, negative numbers can be represented implicitly. For example, we can assign one half of the range, $0 \leq x \leq \frac{M}{2} - 1$ to represent positive integers and the other half, $\frac{M}{2} \leq x \leq M - 1$, to represent negative integers, that is, M - A = -A. This representation is illustrated in Figure 2-1.



$M-1 \leq x \leq \frac{M}{2}$ negative values; $0 \leq x < \frac{M}{2}$ positive values

FIGURE 2.1 RESIDUE REPRESENTATION OF NEGATIVE NUMBERS.

2.3 ADDITION, SUBTRACTION AND MULTIPLICATION

A unique feature of the residue arithmetic is that the computations for each modulus are performed simultaneously but independently without any carry bit. This allows the computations to be carried out in a single step.

2.3.1 ADDITION

Addition is the most basic of the arithmetic operations. The addition of two numbers is illustrated in Figure 2.2. Residue arithmetic is performed on the residue of each modulus and the N-tuple of residue sums will be the residue representation of the sum of the two numbers. That is, if $(a_1, a_2, \ldots a_N)$ and $(b_1, b_2, \ldots b_N)$ are the residue representation of A and B, then $(|a_1 + b_1|_{m_1}, |a_2 + b_2|_{m_2}, \ldots |a_N + b_N|_{m_N})$ represents the sum A + B. We should note however, that the magnitude of the sum must also be within the range of the residue number system $0 \leq z \leq M - 1$. In a later section, we shall discuss in more details the problem of overflow and its detection.

2.3.2 SUBTRACTION

Subtraction can be performed in a similar manner. An example of the subtraction operation is given in Figure 2.3(a). An alternate method is to first transform the subtractor into its additive inverse and sum it with the subtrahand. The additive inverse $|-K|_{m_i}$ is defined by

$$|k + |-K|_{m_i}|_{m_i} = 1$$

With this transformation, the subtraction operation is converted into an addition operation. That is

$$|A - B|_{m_i} = |A + |-B|_{m_i}|_{m_i}$$

DECIMAL		RESIDUE			
	Mod.	2	3	5	7
41 =		1	2	1	6
+ 28 =	+	0	1	3	0
69		1	0	4	6

FIGURE 2.2 RESIDUE ADDITION.

DECIMAL		RESIDUE			
	Mod.	2	3	5	7
41 =		1	2	1	6
- 28 =	-	0	1	3	0
13 =		1	1	3	6

FIGURE 2.3A DIRECT RESIDUE SUBTRACTION.

	Moduli			
	2	3	5	7
28 =	0	1	3	0
+ |-28| =	0	2	2	0 (additive inverse)
0 =	0	0	0	0

DECIMAL		RESIDUE			
	Mod.	2	3	5	7
41 =		1	2	1	6
+ |-28| =	+	0	2	2	0
13 =		1	1	3	6

FIGURE 2.3B RESIDUE SUBTRACTION BY THE USE OF ADDITIVE INVERSE.

There is a one-to-one correspondence between a residue number $|k|_{m_i}$ and its additive inverse $|-k|_{m_i}$. The transformation can be achieved with the use of table look up or fixed mapping. The concept of mapping will be discussed in the next chapter. A subtraction operation using the additive inverse technique is illustrated in Figure 2-3(b).

2.3.3 MULTIPLICATION

Multiplication can also be performed by operating directly on each modulus as shown in Figure 2.6(a). That is, $(a_1, a_2, \ldots a_n) \cdot (b_1, b_2, \ldots b_N) = (|a_1.b_1|_{m_1}, |a_2.b_2|_{m_2}, \ldots |a_N.b_N|_{m_N})$. Alternatively, a homomorphic approach can be taken.

Let m_i-1 be a prime number. The residues 1, 2, ... m_i-1 then form a cyclic group with respect to multiplication of order m_i-1 and each nonzero integer is a power of a prime interger b. For example, with modulus 5, each nonzero residue is a power of 2. The exponential function, given by Table 2-1(a) establishes a one-to-one correspondence between itself and the nonzero residues. Thus for example

$$|2^2|_5 = 4$$

and as with any exponential function

$$|2^{a+b}|_5 = ||2^a|_5 \cdot |2^b|_5|_5$$

Let us define the inverse function of $|2^k|_5$ as $|\log_2 k|_5$ (although it is not the same as the conventional definition of $\log_2 k$ transformation). Thus,

$$|\log_2 4|_5 = 2$$

The table for $|\log_2 k|_5$ transform can be obtained by simply inverting Table 2.1(a) and rearranging as shown in Table 2.1(b).

Using these transformations, a modulo M_i multiplication operation can be converted into a modulo M_i-1 addition. The homomorphic multiplication process is illustrated in Figure 2.4.

MODULUS 5

k	0	1	2	3
$\lvert 2^k \rvert_5$	1	2	4	3

TABLE 2.1A EXPONENTIAL TRANSFORM.

MOD 5

k	1	2	3	4
$\lvert \log_2 k \rvert_5$	0	1	3	2

TABLE 2.1B LOG-LIKE TRANSFORM.

MOD 2, 3, 5, 7,

$x = 11 = [1, 2, 1, 4] \xrightarrow[\lvert \log_b k \rvert]{m_i} [0, 1, 0, 4]$

$y = 13 = [1, 1, 3, 6] \xrightarrow[\lvert \log_b k \rvert]{m_i} [0, 1, 3, 3]$

$[0, 1, 0, 4] + [0, 1, 3, 3] \text{ Mod } m_i - 1$

$[0, 1, 3, 1]$

$\Downarrow \lvert b^k \rvert_{m_i}$

$x \cdot y = 11 \times 13 = 143 = [1, 2, 3, 3]$

FIGURE 2.4 RESIDUE MULTIPLICATION WITH HOMOMORPHIC APPROACH.

If one of the operators is a multiple of modulus, then the corresponding residue would be zero. However, the $|\log_b k|_{m_i}$ transformation is not defined for the value zero and the homomorphic approach cannot be directly applied. Nevertheless, computation for such cases can proceed by noting that if either the multiplier or multiplicand is zero, the product must also be zero.

2.4 DIVISION AND SCALING

Only integer numbers are represented by residue number system. For additions and multiplications, the sums and products are always integers if both the operands are integers. Such is not the case with the division operation. Even if both the division and dividend are integers, the quotient is generally a fractional value. Division operations in the residue system are therefore much more complex. Depending on the problem involved, we separate the division operations into 3 categories:

1. Division with remainder zero
2. Divisor is a modulus or a product of two or more moduli
3. General division.

Let us first examine the remainder zero case. If the dividend is exactly divisible by the divisor, then the quotient would be an integer and it can therefore be represented unambiguously by residue numbers. Under such a condition, the homomorphic approach employed previously for multiplication can be utilized. With this technique, a modulo m_i division is converted into a m_i-1 subtraction operation.

Once again, the transformation of $|\log_2 0|_{m_i}$ is not defined. With multiplication, this problem is circumvented by noting that

$$|x|_{m_i} \cdot |y|_{m_i} = 0 \text{ if } |x|_{m_1} \text{ or } |y|_{m_2} = 0$$

Such a simple solution however, does not exist for the division operation. First, let us examine the case where only the dividend is a multiple of a modulus. For example, with moduli 2, 3, 5, and 7, to perform

$$55 \div 11 = 5$$

we have,

$$r \equiv (1,1,\underline{0},6) \div (1,2,\underline{1},4) = (1,2,\underline{0},5)$$

For the remainder zero case, the corresponding residue of the quotent would be zero if the residue of the dividend alone is zero. That is

$$|x|_{m_i} \div |y|_{m_i} = 0 \text{ if } |x|_{m_i} = 0$$

On the other hand, if only the divisor is a multiple of a modulus, for example

$$54 \div 5 = 10\,\frac{4}{5}$$

$$r \equiv (0,0,\underline{4},5) \div (1,2,\underline{0}.5) = ?$$

then the remainder zero condition would not be satisfied.

Finally, if both the dividend and divisor are a multiple of a modulus such as the operation

$$55 \div 5 = 11$$

$$r \equiv (1,1,\underline{0},6) \div (1,2,\underline{0},5) = (1,2,\underline{1},4)$$

Then the corresponding residue of the quotient could not be computed directly. The most commonly used technique is to perform the division without the modulus where the residue is zero. That is,

$$(1,1,-,6) \div (1,2,-,5) = (1,2,-,4)$$

We note that the value of the quotient $\frac{x}{y}$ must lie within the range of $0 \leq \frac{x}{y} < \frac{M}{y}$. Thus the quotient can be represented with only moduli 2, 3, and 7 as (1,2,4). To obtain the residue for modulus 5, the extension of base technique can be used. The algorithm for the extension of base will be discussed in a later section.

For the remainder zero case, an alternative division method is the use of the multiplicative inverse. The multiplicative inverse $\left|\frac{1}{Y}\right|_{m_i}$ is defined by

$$\left|\frac{1}{Y}\right|_{m_i} \cdot Y\Big|_{m_i} = 1 \text{ for all } m_i$$

For example, with moduli 4, 5, 7, and 11 and Y = 3 = (3,3,3,3); the multiplicative inverse of Y would be $\left|\frac{1}{Y}\right|_{m_i} = \left|\frac{1}{3}\right|_{m_i} = (3,2,5,4)$. Note that

$$\left|\frac{1}{3}\right|_{m_i} = (3,2,5,4)$$

$$\frac{x\ 3}{1} = \frac{(3,3,3,3)}{(1,1,1,1)}$$

The division operation can now be performed as a multiplication. For example, with moduli 4,5,7, and 11,

$$18 \div 3 = (2,3,4,7) \div (3,3,3,3)$$

$$= (2,3,4,7) \times (3,2,5,4)$$

$$= (2,1,6,6) = 6$$

The multiplicative inverse does not exist whenever one of the residue is zero. Thus, the multiplicative inverse technique cannot be used directly if the divisor is a multiple of a modulus. The situation is very similar to that encountered when using the homomorphic technique. The discussion presented earlier for the homomorphic approach would also apply here. If only the residue of dividend is zero, for the remainder zero case the residue for the quotient would also be zero. If the residues for both the divisor and dividend are zero, the division can proceed while ignoring the corresponding modulus. The residue for this modulus is obtained later by using the extension of base technique.

Remainder zero represents a very limited case of division that by itself would not have any practical importance. However, it can be extended to the case where the divisor is a modulus or a product of two or more moduli. We first note that division in a fixed radix system can be implemented very easily if the divisor is a power of the radix or base. For example

with base 10, $12340 \div 10 = 1234$
with base 2, $10110 \div 10 = 1011$

We see that the quotient can be obtained by simply shifting the dividend by an amount specified by the divisor. Although such a simple procedure cannot be applied for the residue system, it is not surprising that the case of the divisor being a modulus would also facilitate the division operation.

For a general division operation, it can be expressed as

$$\frac{x}{y} = \left[\frac{x}{y}\right] + |x|_y$$

With the residue number system, only the integer part of the quotient can be represented. Let us examine the case where y is a modulus,

that is $y = m_k$. The residue representation of x for modulus m_k, $|x|_{m_k} = |x|_y$, would be equal to the remainder of the division operation $\frac{x}{y}$. Thus, if we subtract x by $|x|_y$, the difference would always be exactly divisible by y. The division operation

$$\left|\frac{x - |x|_5}{y}\right|_y$$

can be performed using either the homomorphic approach or the multiplicative inverse technique. For example, with moduli 2,3,5,and 7, to perform $46 \div 5 = 9 + \frac{1}{5}$, the dividend is first subtracted by $|46|_5 = 1$. That is,

$$\begin{aligned} 46 &= (0, 1, 1, 4) \\ -\ |46|_5 &= (1, 1, 1, 1) \\ \hline 46 - |46|_5 &= (1, 0, 0, 3) \end{aligned}$$

The difference is then divided using multiplicative inverse technique

$$(1, 0, 0, 3) \div (1, 2, 0, 5)$$

$$= (1, 0, 0, 3) \times (1, 2, -, 3)$$

$$= (1, 0, -, 2)$$

Since the residue of the divisor for modulus 5 is zero, the division is performed without modulus 5. The residue for modulus 5 is obtained from the values of the other moduli by the extension of the base. We then obtain

$$(1, 0, -, 2) = (1, 0, 4, 2) = 9 = (\frac{46}{5})$$

Next, we shall consider the case where the divisor is the product of two moduli, that is, $Y = m_k m_\ell$. We note that $\frac{X}{m_k m_\ell}$ is always within the range $\frac{M}{m_k m_\ell}$. The quotient can therefore be represented uniquely without moduli m_k and m_ℓ. The division operation is performed by first subtracting the value represented by the residues of m_k and m_ℓ and dividing the difference without moduli m_k and m_ℓ. For example, to perform $47 \div 15 = 3 + \frac{2}{15}$ with moduli 2, 3, 5, and 7, the divisor 15 is a product of two moduli 3 and 5. The residue representation of 47 for moduli 3, 5 is $(2, 2)_{3,5}$, corresponding to the value of 2. Subtracting 2 from the dividend, we have

$$47 = (1, 2, 2, 5)$$

$$\underline{-(2, 2) = (0, 2, 2, 2)}$$

$$47 - (2,2)_{3,5} = (1, 0, 0, 3)$$

The difference is then divided by 15 without the moduli 3 and 5. That is,

$$(1, 0, 0, 3) \div (1, 0, 0, 1)$$

$$= (1, 0, 0, 3) \times (1, -, -, 1)$$

$$= (1, -, -, 3)$$

Using extension of base, we then obtain the quotient

$$(1, -, -, 3) = (1, 0, 3, 3) = 3 = \left[\frac{47}{15}\right]$$

The condition that the divisor be a modulus or a product of moduli is still quite strict. Nevertheless, this limited division procedure can be very useful, especially in performing scaling operations. As we shall discuss in the next section, overflow and its detection is a serious problem in the residue number system. To avoid

overflow, the operands have to be periodically scaled down. Since the scaling factor needs not be an arbitrary number, we can design the computer system such that the scaling factor is equal to the value of a modulus or a product of moduli.

General division is a difficult operation in residue arithmetic or any other integer arithmetic. It is cumbersome, time consuming, and not very accurate. It is generally wise to avoid applications where general division is required. Fortunately, there are many important applications where the algorithms can be structured in such way that the general division operation can be eliminated. Nevertheless, general division can be performed in the residue system when needed. There are a few algorithms proposed but none of them can be performed without many sequential steps. We shall present in the following one of the proposed algorithms. The complexity is quite typical of the procedures for general divisions.

First, a product of moduli is found such that it approximates the divisor. For example, with moduli 2, 3, 5, and 7, to perform $206 \div 13$, we can use the product of moduli 3 and 5 as the approximated divisor $\tilde{Y}$. The division operation is then performed for $206 \div 15$. Since the division is a product of the moduli, we can proceed with the method we described earlier. 206 is represented as $(0, \underline{2}, \underline{1}, 3)$ and (2, 1) modulo 3, 5 corresponds to 11. The divident is then subtracted by 11. That is,

$$(206 - 11) = (0, 2, 1, 3) - (1, 2, 1, 4)$$

$$= (1, 0, 0, 6)$$

$(1, 0, 0, 6)$ is exactly divisible by $\tilde{Y}$, and we can perform the division without moduli 3 and 5,

$$(1, 0, 0, 6) \div (1, 0, 0, 1) = (1, -, -, 6).$$

Using the extension of base technique, we obtain

$$(1, -, -, 6) = (1, 1, 3, 6) = 13$$

The dividend x is then subtracted by $Y\left(\frac{x}{\tilde{y}}\right)$ and the difference is denoted as X'. That is,

$$206 - 13 \times 13 = 37 = x'$$

or

$$(0, 2, 1, 3) - (1, 1, 3, 6)(1, 1, 3, 6) = (1, 1, 2, 2) = x'$$

Using the same procedure the values of $\left(\frac{x'}{\tilde{y}}\right)$, $\left(\frac{x''}{\tilde{y}}\right)\ldots\left(\frac{x^n}{\tilde{y}}\right)$ are recursively computed until

$$\left(\frac{x^n}{y}\right) = 0,$$

then

$$\left(\frac{x}{y}\right) = \left(\frac{x}{\tilde{y}}\right) + \left(\frac{x'}{\tilde{Y}}\right) + \ldots \left(\frac{x^n}{\tilde{y}}\right).$$

For our example

$$\left(\frac{x'}{\tilde{y}}\right) = \left(\frac{37}{15}\right) = 2$$

and

$$x' - Y\left(\frac{x'}{\tilde{y}}\right) = 37 - 13 \times 2 = 11 = x''.$$

Since

$$\left(\frac{x''}{\tilde{y}}\right) = \left(\frac{11}{15}\right) = 0,$$

we have for the round-off quotient,

$$\left[\frac{x}{y}\right] = 13 + 2 + 0 = 15$$

The result obtained $\left[\frac{x}{y}\right]$, corresponds to the integer part of the quotient. Thus, the result can provide good accuracy only if $x >> y$, such that $\left[\frac{x}{y}\right] >> |x|_y$. It may not be the case in general discussion. For example, with $15 \div 8$, the difference between the exact quotient 1-7/8, and the rounded off value 1, is almost 50%. On the other hand, it is generally true that $x >> y$ in scaling operations. Thus, scaling can be performed without severely affecting the computation accuracy.

2.5 ENCODING AND DECODING

Before residue arithmetic can be performed, the operands must first be converted into the residue system. After the computations are completed, the output must also be converted from its residue representation to a number system that is recognizable to a human operator or conventional machine.

2.5.1 ENCODING

The encoding process is in general fairly simple. One may of course, obtain the residue number directly from the relationship

$$r_i = |x|_{m_i} = x - \left[\frac{x}{m_i}\right] m_i$$

However, such a conversion procedure would require the use of non-residue arithmetic. In order to allow a residue computer to perform the encoding, an alternate approach can be used. For example, if the number is originally in fixed radix formed, it can be written as

$$x = a_n b^n + a_{n-1} b^{n-1} + \ldots a_o b^o$$

where a_n is the coefficient and b is the radix. The coefficients a_i and weights b^i can be converted into the residue representations A_i, B^i using table look up. The residue representation of the number x can then be obtained by performing the sum of the product with residue arithmetic. That is,

$$r_i = |x|_{m_i} = |A_n B^{\lambda} + A_{n-1} B^{n-1} + \ldots\ldots A_o B^o|_{m_i}$$

In the example above, the encoding is performed from a fixed radix representation. However, the same approach can be applied equally well for the encoding from a mixed radix number. Encoding will be discussed again in better details in Chapter 4. Specific encoding procedures and implementation techniques will also be presented.

2.5.2 DECODING

The earliest technique for the decoding of a residue number was introduced by Sun Tsu in the first century AD and later formulated by K. F. Gauss in the nineteenth century[7]. The resulting theorem is generally referred to as the Chinese Remainder Theorem. The theorem states that an integer within the range of 0 and M-1 can be written as

$$x = \left| \sum_{i=1}^{N} \hat{m}_i \left| \frac{r_i}{\hat{m}_i} \right|_{m_i} \right|_{m_i}$$

where

$$M = \prod_{j=1}^{N} m_j$$

and

$$\hat{m}_i = \frac{M}{m_i}$$

While the Chinese Remainder Theorem provides one of the simplest method of decoding a residue number, it requires a summation operation that must be performed outside the residue number system. The residue computer itself therefore cannot be used for the decoding procedure. On the other hand, the algorithm for the residue to mixed radix conversion can be computed completely with residue arithmetic. This would allow the decoding to be performed by the residue computer at its system throughput rate. The residue to mixed radix conversion process will be discussed in details in the next section. In Chapter 4, the implementation technique for decoding will be presented.

2.6 RESIDUE TO MIXED RADIX CONVERSION AND EXTENSION OF BASE

In this section, the algorithm for converting a residue number to its equivalence in the mixed radix form will be discussed. This conversion process is singled out for discussion because of its importance not only in the final decoding of the output, but also in performing condition checks such as overflow detection, magnitude comparison and error detection. The conversion algorithm can also be modified to perform the extension of base which is an integral part of the division or scaling operations.

2.6.1 RESIDUE TO MIXED RADIX CONVERSION

A mixed radix system is composed of a set of radices m_1, m_2, m_3, ... m_n; and a number is represented by

$$(a_1, a_2, a_3, \ldots a_n)$$

such that

$$x = \sum_{n=1}^{N} \left(a_n \prod_{i=1}^{n-1} m_i\right)$$

$$= a_1 + a_2 m_1 + a_3 m_1 m_2 + \ldots a_N \prod_{i=1}^{N-1} m_i$$

If the radices of a mixed radix system are chosen such that they are identical to the set of moduli of a residue system, then the two system is said to be associated. These associated number systems will have the same range of integers that can be represented uniquely. More importantly, the algorithm for the residue to mixed radix conversion can be performed using residue arithmetic. This would allow the conversion to be performed by the residue computer itself. Because of the potentially high throughput rate an optical residue computer, it is essential that the decoding be performed at the same rate. In Section 4, we shall show that through pipelining, the residue to mixed radix conversion can be performed by the residue computer at the system throughput rate.

The coefficients of the mixed radix number can be obtained by the relationship,

$$a_i = \left|\left\lfloor \frac{x}{m_1 m_2 \cdots m_{i-1}} \right\rfloor\right|_{m_i}$$

The algorithm is best demonstrated with an example. Let us assume that a residue system with moduli 3, 4, 5, and 7 is used and the residue numbers are to be converted to an associated mixed radix system. We begin by noting that $r_1 = a_1$. The residue r_1 (or a_1) is then subtracted from x and the difference would be divisible by m_1. Division using the multiplicative inverse approach can be performed and the quotient for modulus m_2 would correspond to the coefficient a_2. The entire procedure is illustrated in Figure 2.5.

2.6.2 EXTENSION OF BASE

As discussed previously, in performing division operations, the extension of base is necessary to obtain the residue of the modulus where the residue is zero for the divisor. The extension of base can be performed by modifying the residue to mixed radis conversion process. For example, to extend the base of a residue number represented by moduli $m_1 m_2 \ldots m_{N+1}$, we can let the coefficient a_{N+1} of the mixed-radix representation be zero since the residue number is within the range

$$M = \prod_{i=1}^{N} m_i$$

With this prior knowledge, the residue for modulus m_{N+1} can be obtained by the residue to mixed radix conversion process as illustrated in Figure 2.6.

2.7 OVERFLOW DETECTION, MAGNITUDE COMPARISON AND ERROR DETECTION

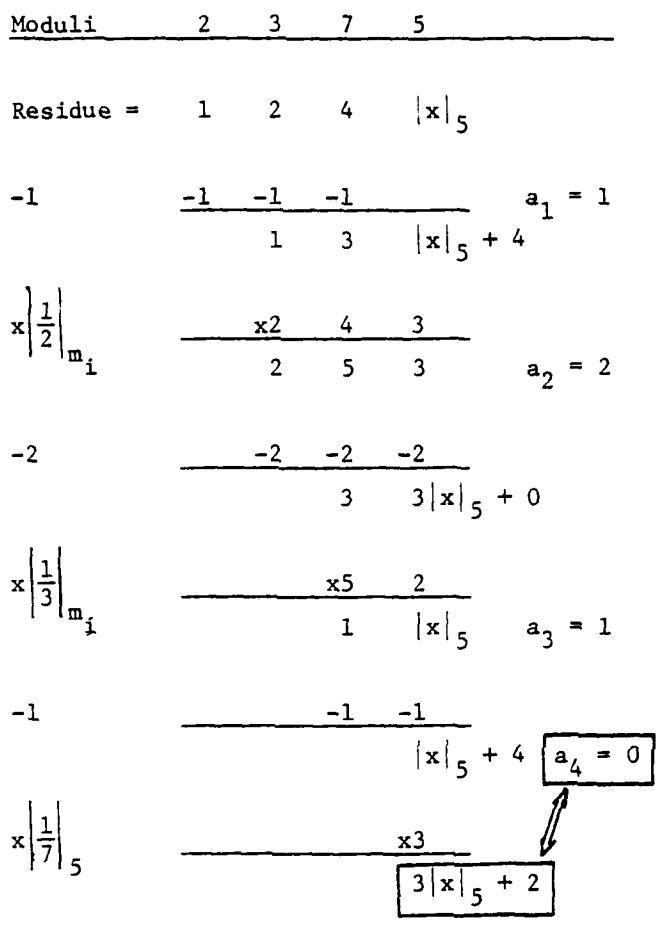
Overflow detection is a trivial problem with weighted number systems. With the residue number system however, overflow detection is not so automatic. First of all, the magnitude of a residue number is not evident from the values of its residues. Secondly, the residue number system is cyclic over the range

$$M = \prod_{i=1}^{N} m_i$$

Moduli	3	4	5	7	
$x = 389$	[2]	1	4	4	$[a_1 = r_1 = 2]$
$(-a_1) - 2$	-2	-2	-2	-2	
		3	3	3	
$\left(x\left\|\frac{1}{m_i}\right\|_{m_i}\right) x \left\|\frac{1}{3}\right\|_{m_i}$		x3	3	5	
		[1]	4	3	$[a_2 = 1]$
$(-a_2) - 1$		-1	-1	-1	
			3	2	
$\left(x\left\|\frac{1}{m_2}\right\|_{m_i}\right) x \left\|\frac{1}{4}\right\|_{m_i}$			x4	2	
			[2]	4	$[a_3 = 2]$
$(-a_3) - 2$			-2	-2	
				2	
$\left(x\left\|\frac{1}{m_3}\right\|_{m_i}\right) x \left\|\frac{1}{5}\right\|_{m_i}$				x3	
				[6]	$[a_4 = 6]$

$$x = 6(3 \times 4 \times 5) + 2(3 \times 4) + 1(3) + 2 = 389$$

Figure 2.5. Residue to Mixed Radix Conversion



$3|x|_5 + 2 = a_4 = 0$

$|x|_5 = 1$

$x = 11 = [1, 2, 4] \Rightarrow [1, 2, 4, 1]$

Mod 2,3,7 Mod 2,3,7,5

Figure 2.6. Extension of Base

That is, the residue representation is the same for integers k, k+M k+2 M, etc. For additive overflow, the problem is a little simpler. By using a range M which is twice as large as the required range $\overline{M}$, we can avoid the situation where the sum of two numbers is larger than M and the residue representation circles back to produce an erroneous answer. Overflow can be detected by converting the sum from the residue system to a weight number system (e.g., mixed radix system). The magnitude is then determined to see if it is in range. For example, if the required range is

$$\pm\overline{M} = \pm \prod_{i=1}^{n} m_i,$$

for overflow detection, moduli m_n, m_{n-1}, ..., m_1, 4 can be used. When the residue number is converted into the mixed radix system it becomes

$$x = A_o(1) + A_1(m_n) + A_2(m_n \cdot m_{n-1})$$

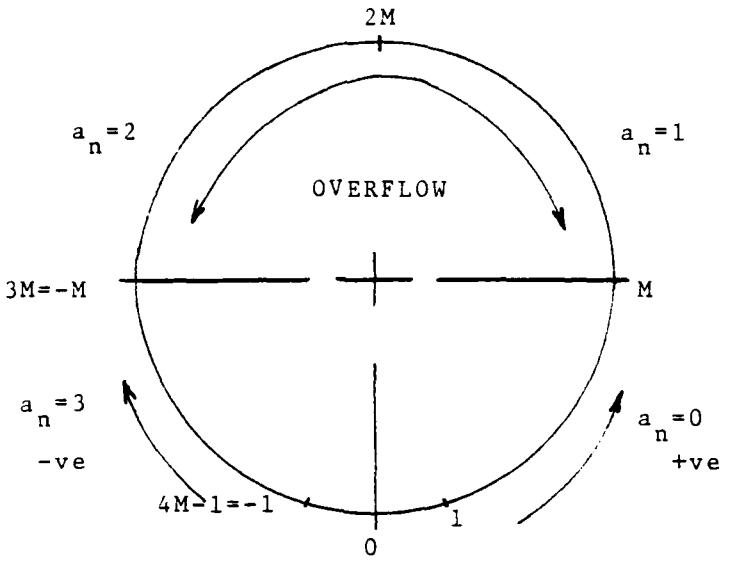
$$+ \ldots A_n(m_n \cdot m_{n-1} \cdots m_2) + A_{n+1}(\overline{M})$$

The value of A_{n+1} would provide an indication of the sign and range of the residue number as illustrated in Figure 2.7.

Multiplicative overflow cannot be detected by simply extending the range. In order to do that, the range would have to be MM, which is impractically large. To detect multiplicative overflow, the magnitude of the multiplicand and the multiplier must be checked before the multiplication is performed.

One possible technique is to convert both operands into the mixed radix form with an order of M_1, M_2, ..., M_K, M_{K+1}, ..., M_N

where $$M_1, M_2, \ldots, M_K \cong \sqrt{\prod_{i=1}^{N} M_i}$$



$a_n=o \longrightarrow$ +ve

$a_n=1$ or $2 \longrightarrow$ overflow

$a_n=3 \longrightarrow$ -ve.

FIGURE 2.7 OVERFLOW DETECTION WITH RESIDUE TO MIXED RADIX CONVERSION.

For example, with MOD 2, 3, 5, and 7; the range M is equal to $\prod_{i=1}^{4} m_i = 210$; the conversion into mixed radix form is performed in the order of 3, 5, 2, 7 since $3 \times 5 \cong \sqrt{210}$.

Let us denote operands A and B in the mixed radix form as

$$\left[A_3, A_5, A_2, A_7\right], \left[b_3, b_5, b_2, b_7\right]$$

No overflow will occur for A × B if one of the following conditions is met:

1. $A_2 = A_7 = b_2 = b_7 = 0$
2. $A_2 = A_7 = b_7 = 0$ and $A_5 \leq 2$
3. $A_7 = b_2 = b_7 = 0$ and $b_5 \leq 2$
4. $A_7 = A_2 = A_5 = 0$ and $b_7 = 1$
5. $b_7 = b_2 = b_5 = 0$ and $A_7 = 1$

Example: Mod 2, 3, 5, 7

$$A \times B = 7 \times 80 = \left[1, 1, 2, 0\right] \times \left[0, 2, 0, 3\right]$$

After the conversion into the mixed radix form, we have

$$\left[1, 2, 0, 0\right] \times \left[2, 1, 1, 1\right]$$

$$A_3, A_5, A_2, A_7 \qquad b_3, b_5, b_2, b_7$$

Since none of the conditions listed above is satisfied, overflow occurs.

Since overflow detection can be detected much easier for a weighted number system, another approach is to check for possible overflow while the input is still in the binary form (from A/D converter).

Let us assume that the range of the residue computer is M bits and we would like to check if the product of two numbers will be within the range M. Let us define A and B as the sizes in bits of the two operands to be multiplied. For example

$$\underbrace{00101001}_{A=6\text{ bits}} \times \underbrace{00110110}_{B=6\text{ bits}}$$

Thus, if $[A + B] - M < 0$, product in range
> 0, overflow

For the above example, if $M = 8$ bits, then $[A + B] - M = 4 > 0$ and overflow will occur. And indeed,

$$00101001 \times 00110110 = \underbrace{100010100110}_{> M\text{ (overflow)}}$$

The product is larger than $M = 8$ bits. In order to avoid overflow, the numbers must first be scaled down by an appropriate factor. This can be done by shifting the two numbers to be multiplied by a total of $[A + B] - M$ bits.

Using the same example, $[A + B] - M$, we should shift the two numbers by a total of 4 bits, (e.g., 2 bits each). We now have

$$00101001\ [41] \times 00110110\ [54]$$

$$00001010\ [10] \times 00001101\ [13]$$

$$= \underbrace{10000010}_{\leq M\text{ (in range)}}\ [130]$$

The product is within range of M = 8 bits. To produce the correct result, the product has to be scaled back up by the same factor.after it is decoded from the residue system. This can be achieved very easily by shifting the product by [A + B] - M bits. Thus

$$10000010 \rightarrow 100000100000 = [2080]$$

Comparing this result with the exact answer 1000101001100 [2214], we see that the round off error is 6%.

There is no actual calculation involved with this technique. It requires only simple logic controls and it can therefore be executed at very high speed.

Magnitude comparison is essentially a subtraction and sign determination procedure.

To compare the magnitudes of two numbers A and B,

if			
	$A \geq 0$ and	$\lvert B - A \rvert_M \geq 0$	then $A \leq B$
	$A \geq 0$	$\lvert B - A \rvert_M < 0$	$B < A$
	$A < 0$	$\lvert B - A \rvert_M \geq 0$	$A \leq B$
	$A < 0$	$\lvert B - A \rvert_M < 0$	$A < B$
	$A < 0$	$\lvert B - A \rvert_M < 0$	$B > A$

To determine if A and $\lvert B - A \rvert_M$ is positive or negative values, they can be converted into the mixed radix form. Using the sign representation discussed previously. A is positive if $0 \leq A \leq M/2-1$ and A is negative if $\frac{M}{2} \leq A \leq M-1$.

Another approach is to simply convert both A and B into the mixed radix form and compare their magnitudes digit by digit, starting from the most significant digit. For example, with moduli 4, 5, 7 to compare 100 and 65, we have

decimal	=	100	65
residue	=	[0,0,2]	[1,0,2]
mixed radix	=	5,0,0	3,1,1

Since 5 > 3

$$[0,0,2] > [1,0,2]$$

There are certain redundancy associated with the residue code. It is not unlike holography where an image point is coded into a multiple of points in the hologram. Losing some bits of information in the hologram would not result in the loss of a part of the image. It only loses the precision with which the image points can be determined. The same is true with the residue code. Losing one residue number would not produce a complete loss of the coded value. Instead, there will be a multiple of possible values within the range M that are represented by the remaining residues, analogous to a loss in image resolution for a degraded hologram.

The above discussion applies only to the case where residue number is missing, or known to be in error and discarded. If the output produces an erroneous but legitimate result, a way must be devised to detect it.

One method is to use an extra modulus to be used as a check code. This extra modulus should be larger than the rest of the moduli in the numbers system. For example, with moduli 2, 3, 5 and a range of 30, we can add an extra modulus of 7. If any digit is in error, the erroneous residue number would have a magnitude beyond that of the range M = 30. This can be checked by converting the residue number into the mixed radix form. For example, for an output of 24 = [1, 0, 4, 3], an error occurs in the residue of modulus 2 and the output becomes [0, 0, 4, 3]. [0, 0, 4, 3] corresponds to 129 and it is therefore larger than the range M = 30. An error in the residue arithmetic is thus indicated.

3
PHYSICAL REPRESENTATION OF RESIDUE NUMBER SYSTEM AND IMPLEMENTATION APPROACHES

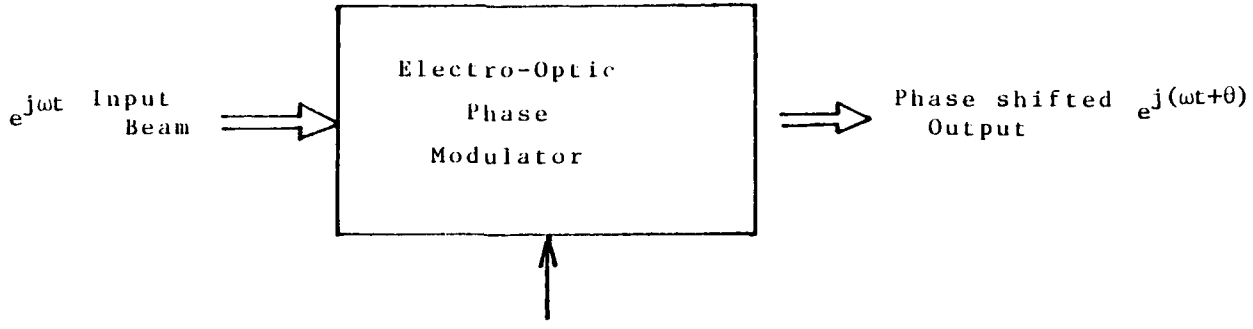
3.1 INTRODUCTION

Fundamental to the optical implementation of a numerical processor is the use of devices which provide numerical control of a light beam or wave. In this section, the use of various physical properties of a light beam to represent residue numbers will be discussed together with some possible implementation concepts for performing residue arithmetic.

Among the physical properties that may be considered are phase, polarization, intensity and spatial position. Phase and polarization are of special interest because of their cyclic properties. In increasing the phase or polarization angle of a light beam, a modulo 2π addition is in effect performed. The use of spatial positions to represent residue numbers has the advantage of allowing the residue numbers to be represented in binary form with m_i discrete positions. The discussion in this section will center on the unique features of these two distinctively different concepts. The possibility of combining the two concepts will also be discussed.

3.2 PHYSICAL REPRESENTATION

Consider the control of light wave phase as a light beam passes through an electro-optic modulator depicted functionally in Figure 3.1. Since the phase of the light wave is inherently cyclic modulo 2π, then by providing control of the phase shift in increments of Δ where $\Delta = 2\pi/m$ and m the desired modulus, the phase of the emergent light wave can serve as a residue number representation. For example, for modulus 5, we make $\Delta = 2\pi/5$. Changing the phase incrementally, we progress through Δ, 2Δ, 3Δ, and 4Δ and then start to repeat modulo 2π such that we have an equivalent representation between 0 and 5Δ, 2Δ and 6Δ, etc. With an input (control voltage) that is continuous rather than quantized, the optical phase shift modulator device may be designed to provide a quantized response, analogous to



$\theta = n\Delta$

$\Delta = 2\pi / m$

FIGURE 3.1 A GENERALIZED CYCLIC DEVICE.

approaches under development with polarization modulation schemes[8,9]. Otherwise, quantization must be provided at the point of detection or in the applied control voltage itself.

In addition to electro-optic devices, other devices available are based on acoustic-optics, thermo-optics, and material deformation (optical path length modulation) for the control of the phase of a light wave.

Rather than altering the phase of light wave, the polarization angle which is also inherently cyclic can be used for residue number representation[1,8]. The choice of approaches for realizing quantized control are similar to that discussed above for phase control.

A different representation approach is the use of spatial positions[1,10,11]. Each residue number can be represented by a different spatial position. Since modulus m_i is generally not very large, all the possible residue numbers can be represented by distinct resolvable positions within a relatively small area. The use of spatial positions for the representation of residue numbers has the advantage of retaining the low error rate inherent in a binary machine. It is particularly important to a residue computer since error checking is more difficult to implement. Unlike phase or polarization angle, spatial position is not inherently cyclic. Nevertheless, the cyclic characteristic can be inserted in the implementation approach.

Other physical representations such as intensity levels or frequencies can also be used. These representations, much like spatial positions, are not inherently cyclic. However, intensity and frequency representations do not possess the advantages of a binary representation such as spatial position. Thus, the representation by intensity or frequency would encompass the weaknesses of the phase and position representations but not their advantages.

3.3 IMPLEMENTATION APPROACHES

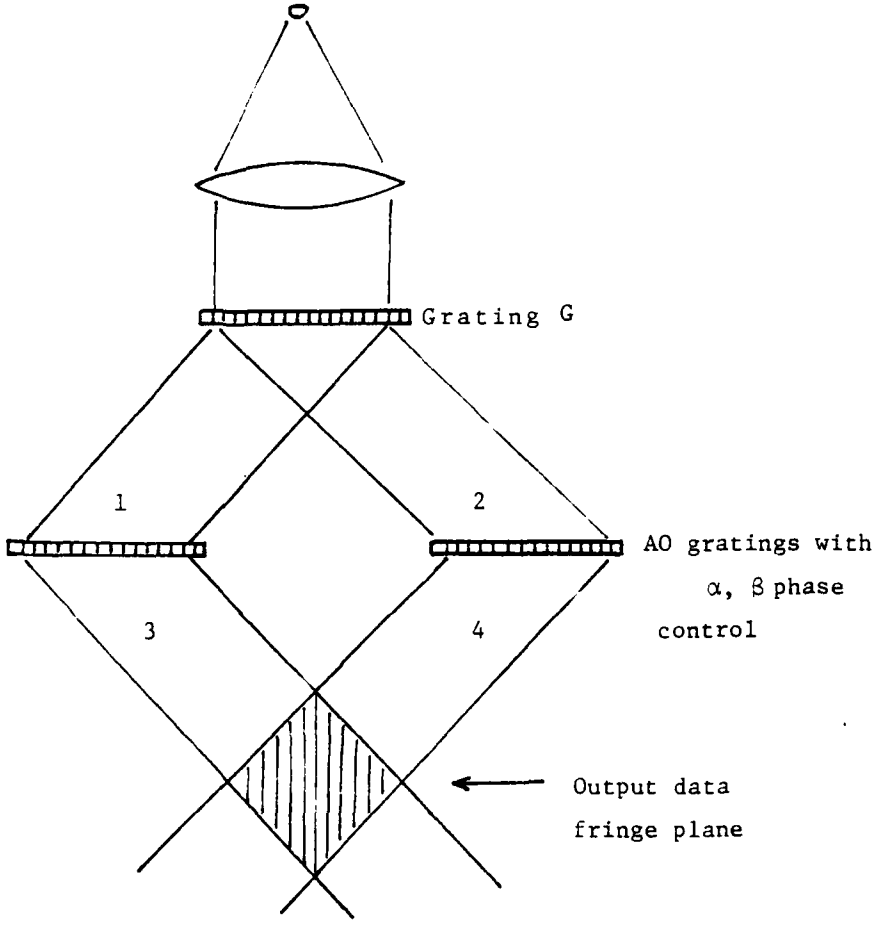
There are two important characteristic inherent in residue arithmetic. In order to implement a residue computer efficiently, these two unique features should be taken advantage of. First of all, the residue numbers are cyclic over the range of modulus m_i. Physical phenomena such as phase and polarization angle that are naturally cyclic can be used to perform operations in residue arithmetic. Other physical representations may also be used if the proper design of a controlling device can be made to produce the cyclic behavior.

Another characteristic of residue arithmetic is the decomposition of a computation with a range of M possible output values into N parts, each having only m_i possible results. This feature makes the use of lookup tables for computation feasible. A table look up is essentially a mapping operation. For example, to perform A x B = C, the operation can be looked upon as the mapping of a set of number A into a set of numbers C. Computations can therefore be performed by a physical implementation of the mapping operation.

In the following sections, the cyclic and mapping approaches will be examined. The discussion will center on the general characteristics of these approaches. In a later section, a more specific design will be presented using the mapping approach.

3.3.1 CYCLIC IMPLEMENTATION APPROACH

As an example of cyclic implementation, we take the case of spatial phase modulation which can be implemented with such devices as acousto-optic spatial phase modulators[12]. We start with the basic set of components shown in Figure 3.2, which consist of two modulators and the means for introducing collimated light waves into each of them. The collimated beams 1 and 2 originate from a laser diode light source directed through a collimating lens and a beamsplitter grating G. This arrangement serves as an interferometer which provides an interference or fringe pattern at its output[13]. It



AO Index Modulation: cos (ωx + α); cos (ωx + β)

Output Light Intensity: ~1 + cos [ωx + (α + β)]

FIGURE 3.2 CYCLIC ADDITION WITH GRATING INTERFEROMETER.

will operate with light sources of modest coherence. The spatial frequency (carrier) in the modulators is twice the grating frequency in G. If the modulators have a sinusoidal spatial modulation of optical index along their length (X-dimension) of the form $\cos(\omega x + \alpha)$ and $\cos(\omega x + \beta)$, then the diffracted first order light waves 3 and 4 can be written as

$$e^{-j(\omega+\alpha)} \text{ and } e^{+j(\omega x + \beta)}$$

Assuming for convenience that these waves have unity amplitude, then the interference or fringe pattern at the detector plane will be of the form

$$1 + \cos[\omega x + (\alpha + \beta)]$$

Thus the phase of the fringe pattern output is the sum of the input or modulator phase α and β. Phases and α and β would be entered into the acousto-optic modulators as equivalent residue numbers for a particular modulus m_i. Since the accumulated output phase is cyclic, the output sum will have the desired residue property of being cyclic modulo m_i.

Rather than using a detector at the output, two other possibilities exist. An optical transducer or memory may be used at the output plane which records the sinusoidal fringe pattern and then acts as a diffraction grating containing the output phase $(\alpha + \beta)$. When illuminated, it would serve to readout the summation data $(\alpha + \beta)$ as the phase of the diffracted output beams for use in another cascaded computing element possibly of the same type. A second possibility for handling the output avoids the use of a detector or a transducer by simply allowing the output waves 3 and 4 to continue and become inputs to another set of acousto-optic modulator elements as shown in Figure 3.3. This

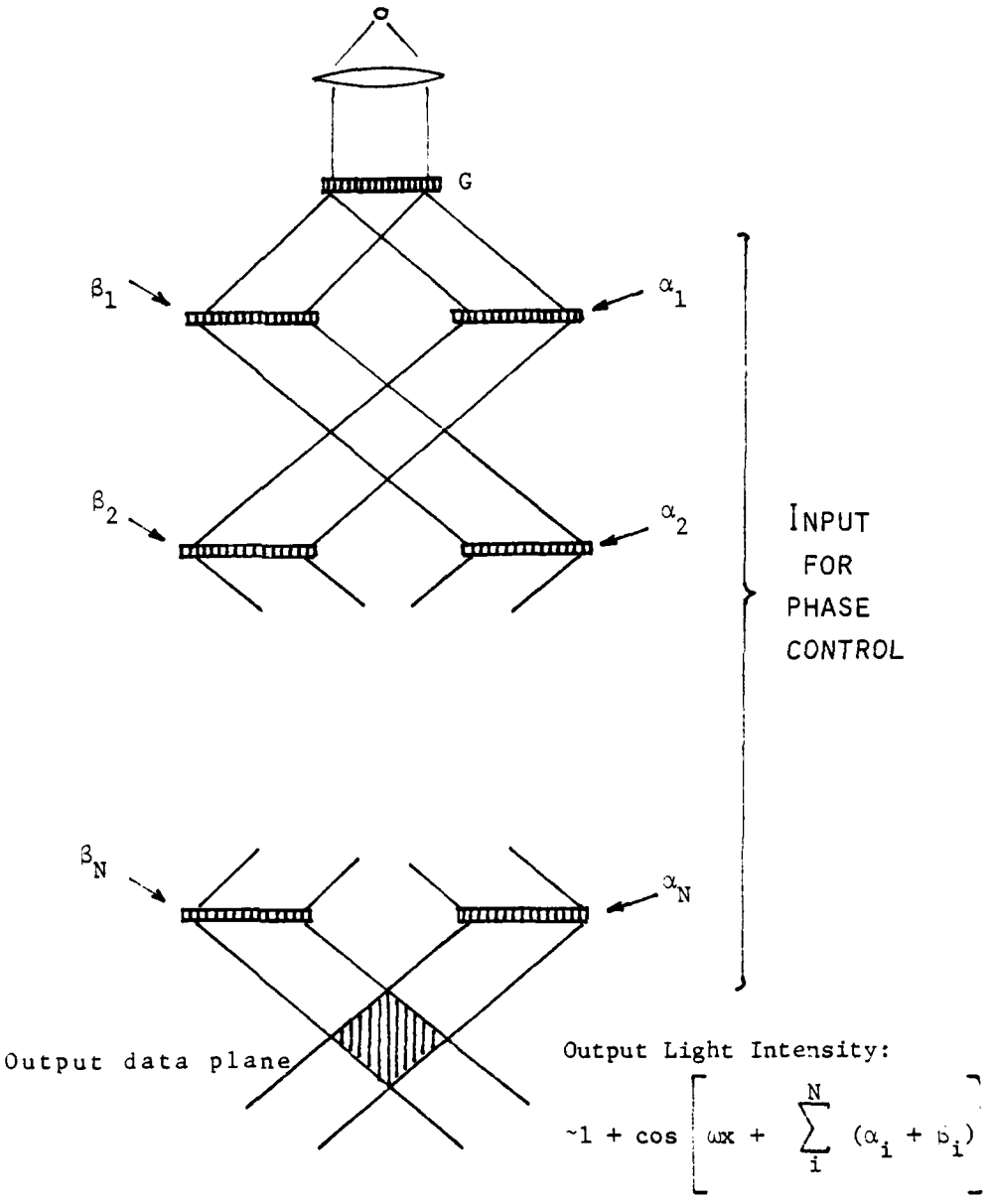


FIGURE 3.3 CASCADING GRATINGS FOR LONG SEQUENTIAL ADDITION.

figure shows a succession of such cascaded modulators. The phase of frange pattern output for the set would be the accumulated sum $\Sigma(\alpha_1 + \beta_1)$, i.e., the output fringe pattern is of the form

$$\cos\left[\omega_x + \sum_{i}^{N} (\alpha_i + \beta_i)\right]$$

With this device, we can realize a computing module capable of addition subtraction and multiplication (through successive addition). For addition or subtraction of two residue numbers, we need only two modulators. With multiplication, the number of modulator elements in this type of computing module must equal the largest multiplier which, for modulus m_i, will be $m_i - 1$. Multiplying two numbers X x Y is done by entering a phase $\alpha = \beta = X$ in all modulator elements and having the total number of modulators equal to Y.

The time required to perform the single summation $\alpha + \beta$ is quite short, being simply the propagation time from the AO cell to the output fringe detection plane. It could be only a few picoseconds for small integrated optic configurations. Clearly, however, overall cycle time of such a unit is the characteristic of importance and it is comprised of the set time of an AO device, the light propagation time and the output fringe phase detection time. At the present state-of-the-art, the AO cell set time capability is in the range 0.1 to 10 μsec which is quite modest for the computing operations of interest here. Another problem with this approach is the limited capability of present devices in performing the phase detection of the output fringe pattern with very high speed and accuracy.

Using the inherent cyclic characteristics of phase or polarization angle, the addition operation can be performed very naturally. However, extending the implementation approach to multiplication and other more complicated functions such as x^n cannot be easily achieved. To perform multiplication, one approach is to sequentially add the multiplicand by itself n times for a xn operation. For large moduli, such an implementation would be tedious. Moreover, a maximum of

m_{i-1} addition operation have to be performed and the quantization errors would accumulate as each addition operation performed. To avoid such accumulation of errors, it would be necessary to use a cyclic device with multistable states characteristic. While there have been some success in producing devices with a limited number of stable states, the problem of extending it to many stable states remains.

3.3.2 MAPPING IMPLEMENTATION APPROACH

The use of spatial position for the representation of residue numbers has the advantage of retaining the low error rate inherent in a binary machine. It is particularly important to the residue number system since error checking is more difficult to implement.

One of the unique features of residue arithmetic is the breaking up of a computation with M possible answers into N parts, each having only m_i possible answers. Residue arithmetic can therefore be implemented with N tables where the m_i possible answers for each modulus can be looked up. Indeed, residue arithmetic has been implemented by electronic computer engineers using precisely this approach. However, the computation rate is limited by the speed with which the table look up can be accomplished (i.e., access time of the memory). Using position representation of residue numbers, this table look up can be implemented very simply as a spatial map. For example, in the operation A x B = C, for each modulus there is a one to one correspondence between the multiplicand A and product C. That is, $|A|_{m_i} \overset{xB}{\Rightarrow} |C|_{m_i}$. As an illustration to perform A x 4 = C modulo 5, we have

A	X4	C
0	→	0
1	→	4
2	→	3
3	→	2
4	→	1

Thus the operation X4 modulo 5 can be implemented as a spatial map as shown in Figure 3.4. The routing in the map can be constructed with electrical wires or optical waveguides. A signal entering the input port at the position corresponding to $|A|_5$ will emerge at the output position which is the spatial representation of the product $|C|_5$. The same concept can be applied to more complex mathematical operations. For example, to evaluate a polynominal $3x^2 + 2x + 4 = z$, we can once again create a map for $|x|_{m_i} \Longrightarrow |z|_{m_i}$. The spatial map for the evaluation of the above polynominal is shown in Figure 3.5 for modulus 5.

The hardwired (fixed) maps presented above can each be used for only one operation. A more flexible approach is to utilize programmable light deflectors to steer the light beam into selectable paths as shown in Figure 3.6. Each of the optical switches has two output ports which are selectable by a control voltage. The switches either allow the entering light beam to pass through undeflected or steer the beam to an alternate path. Since a light beam entering into any of the m_i input port can be deflected into m_i possible output positions, it requires a total of $m_i(m_i-1)$ switches to implement a fully programmable map. The light beam paths may be open or confined (e.g., optical waveguides, fibers, stacked diffraction gratings, etc.). Switching devices such as optical waveguide couplers, acousto-optic diffraction cells and fiber optic couplers may be used for position or path control. The beam paths are discrete and the switches are generally controlled with binary control signals.

Instead of individual two state switches for beam position control, devices having multiple output positions may also be used. Some acousto-optic beam deflector designs for example, can provide over 10^3 discernible output positions which might serve in place of a set of two-position optical switches.

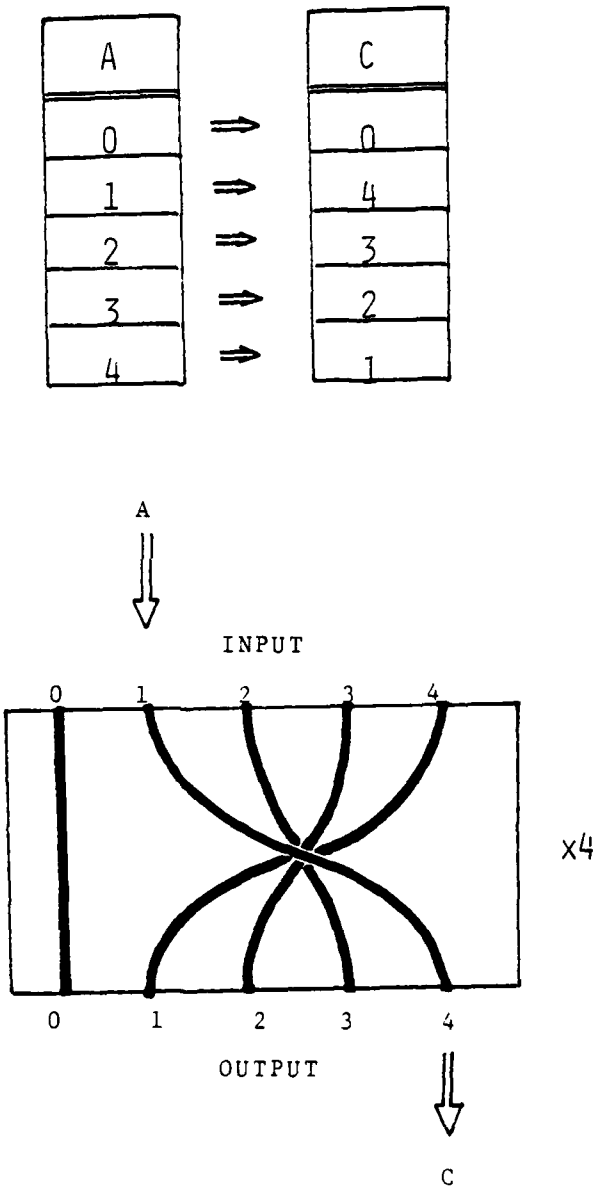
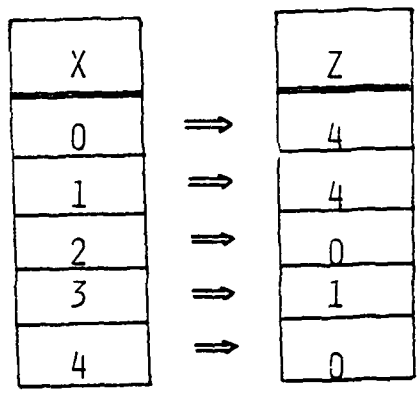
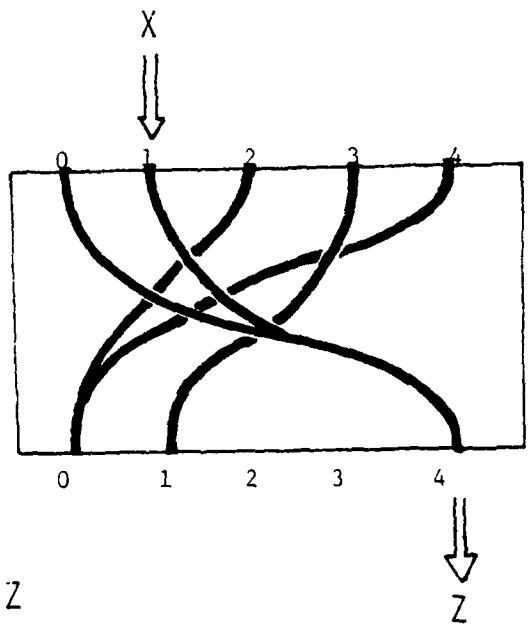


FIGURE 3.4 SPATIAL MAP FOR x4 OPERATION.





$3X^2+2X+4 = Z$

FIGURE 3.5 SPATIAL MAP FOR THE EVALUATION OF A POLYNOMIAL.

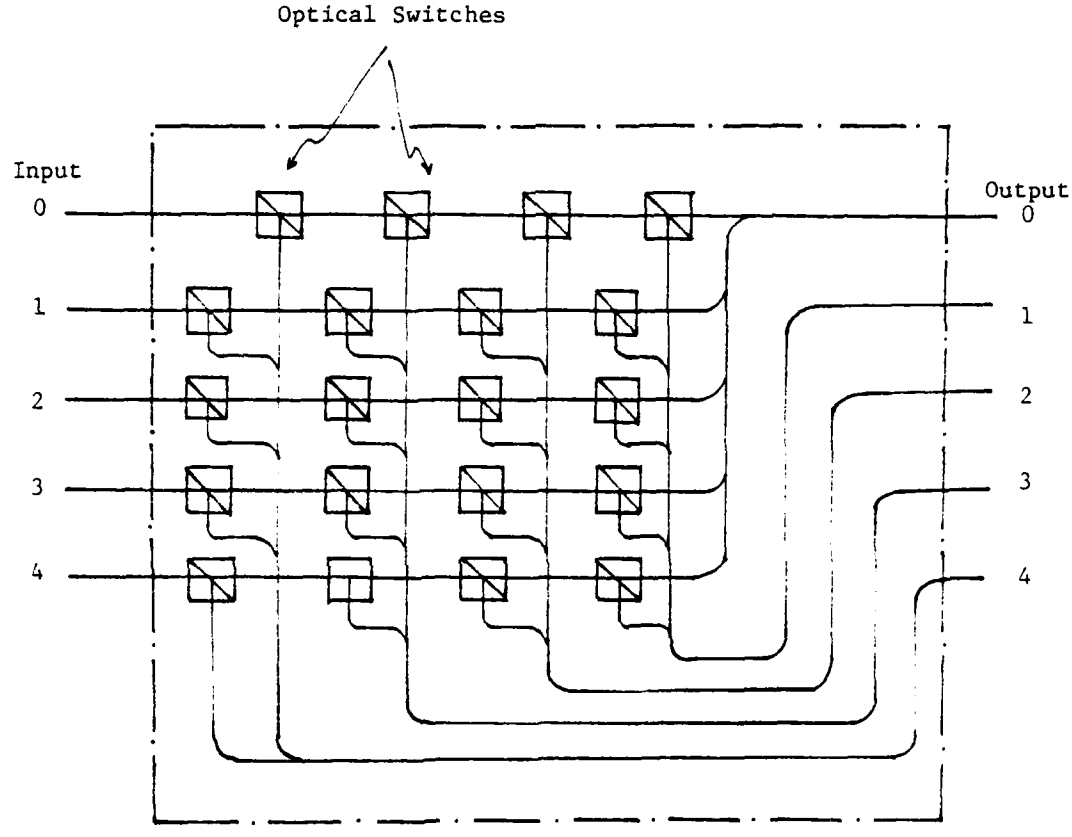


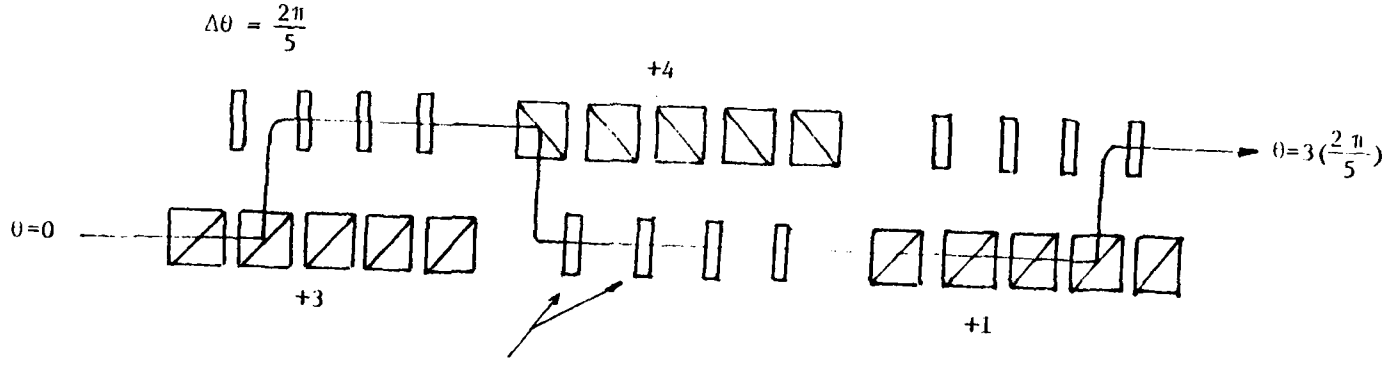
FIGURE 3.6 PROGRAMMABLE MAP IMPLEMENTED WITH LIGHT SWITCHES.

Beside the low probability of error provided by the binary representation, the mapping approach also offers the advantage of flexibility. A computation map can perform a complicated computation procedure (e.g., evaluation of polynominals) as easily as a simple addition operation. While residue addition and subtraction are performed very naturally with the cyclic approach, the concept cannot be extended much beyond addition without the simplicity of the concept being lost.

3.3.3 COMBINATION OF CYCLIC AND MAPPING APPROACHES

In our discussion so far, we have associated the phase and polarization representation exclusively with the cyclic approach and the spatial position representation with the mapping approach. Although it is a natural association, there is no reason why they cannot be used in different combinations. An example of such an implementation is illustrated in Figure 3.7(a). For this case, the phase (or polarization) shift elements are of a fixed and passive type. The amount of phase shift is determined by the light path which is controlled by the states of the optical switches.

Since the incremental phase (or polarization) shifters are fixed, they can be made very accurately. This would reduce the probability of error, allowing a longer sequence of operations to be performed. This approach can also be used in conjunction with a spatial map as shown in Figure 3.7(b). The best of the two approaches are therefore combined using the spatial maps for the more complex operations and the phase shifters for additions. With this implementation, m_i discrete light paths are still required within this device. However, unlike using exclusively the position representation, there is only one light beam position entering or leaving the device.



Fixed phase or polarization shifter of increment $\Delta\theta = 2\pi/5$

FIGURE 3.7A

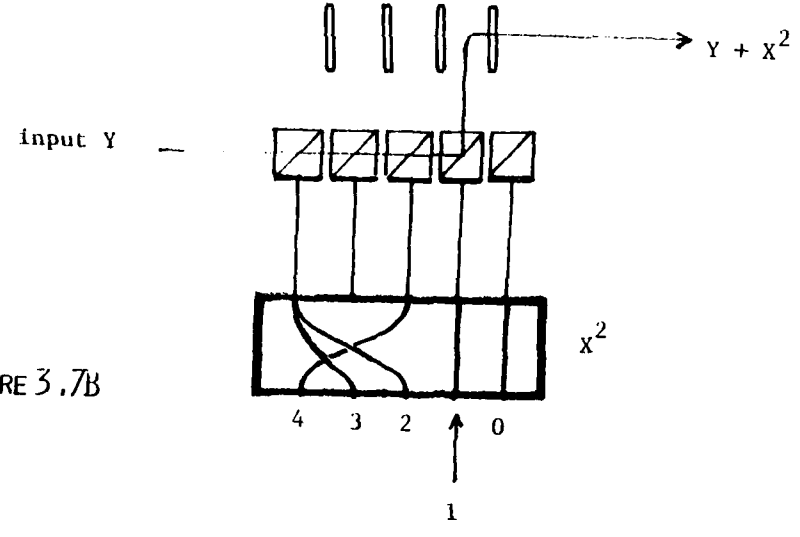


FIGURE 3.7B

COMBINED USE OF CYCLIC AND MAPPING APPROACHES.

Similarly, the spatial position representation can also be used with the cyclic approach. A geared wheel is a perfect example of a device that utilizes position representation which is also inherently cyclic. In fact, some of the earliest attempts in implementing a residue computer made use of geared wheel. In some ways, the characteristics of a geared wheel is ideal for a residue computer. Many of the current efforts in developing hardwares for residue arithmetic with the cyclic approach are actually looking for an electronic or electro-optic equivalence of a geared wheel!

3.6 OPTICAL SWITCHES

To implement the mapping approach, optical switches are used to guide the input light beam through a predetermined path to the appropriate output. There are several possible devices that can be used to construct such a programmable computation map. Acousto optic modulators for example, can deflect a light beam into many resolvable output positions[13]. The use of acousto optic modulators to implement a computation map for the x2 operation is illustrated in Figure 3.8. With the capability of the acousto optic modulators to deflect a light beam to a multiple of output positions, the programmable map can be implemented with less switches than with other devices. However, due to the constraint imposed by the small diffraction angle, the size of the computation map would be relatively large. Moreover, the switching speed of acousto optic modulators is undesirably slow.

An alternative device is the electro optic grating[14]. A grating like electrode structure is used to induce a diffraction grating in the $LiNbO_3$ crystal. The switching speed is significantly faster than the acousto optic deflectors but the light beam can only be deflected to one preset direction. The diffraction efficiency is also quite low, only a small portion of the input light is switched to the desired output position.

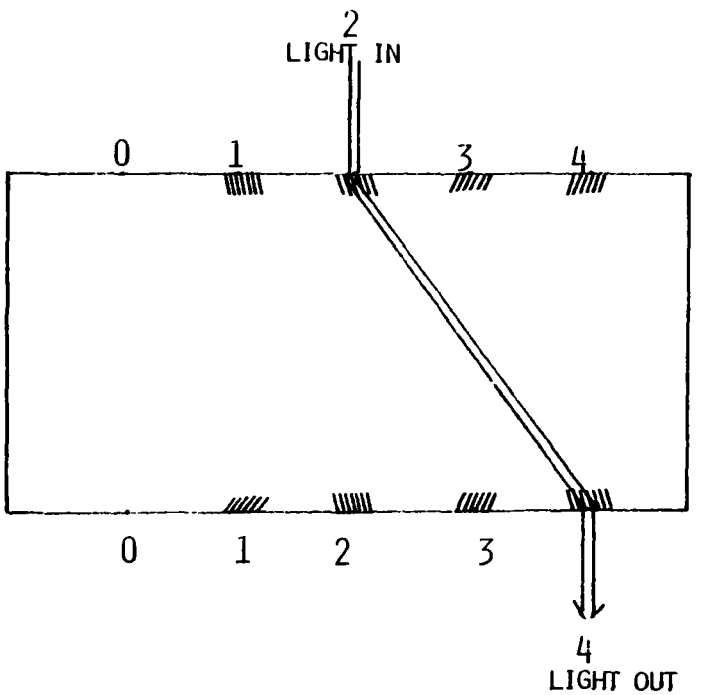
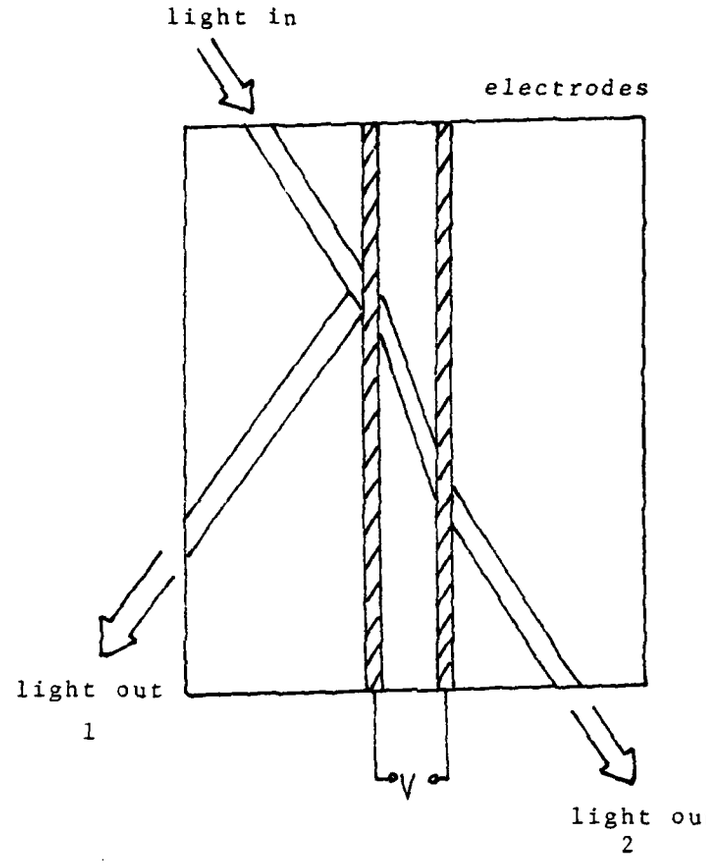
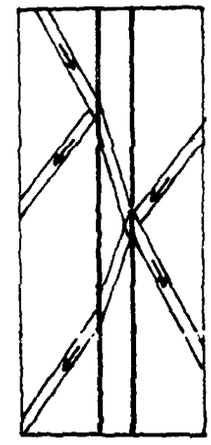


FIGURE 3.8 IMPLEMENTATION OF PROGRAMMABLE MAP WITH AO MODULATORS.





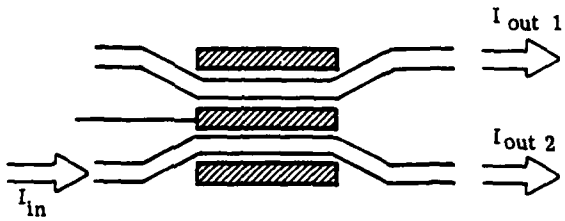
displacement of
transmitted and
reflected beams

FIGURE 3.9 TOTAL INTERNAL REFLECTION OPTICAL SWITCH.

Another electro optic device is the total internal reflection[15] optical switch as shown in Figure 3.9. The electro optic material at the center has a refractive index that is slightly lower than that of the surrounding substrate. The input light beam is injected at an incidence angle that is close to the critical angle. By varying the electrode voltage, the refractive index of the electrode material is changed. The critical angle for total internal reflection will change accordingly such that depending on the electrode voltage, the incident beam will either be reflected or transmitted.

One of the most promising optical switches is the directional waveguide coupler. It is the most versatile of the electro optic switches. A direction coupler is schematically shown in Figure 3.10. Two wave guides are placed physically close to each other such that in the absence of an applied electric field, the wave guides are synchronous. That is, a light wave propagating in one wave guide will be coupled to the adjacent one producing a switch in light path [16-18]. When an appropriate voltage V_T is applied to the electrode, the synchronism between the wave guides is broken and the light propagation will remain in the wave guide originally excited as illustrated in Figure 3.10. The co-directional coupling feature provides a flexibility not obtainable with other types of optical switches. The total internal reflection optical switch cannot be operated under this mode because of the physical displacement between the transmitted and reflected beams as illustrated in Figure 3.9.

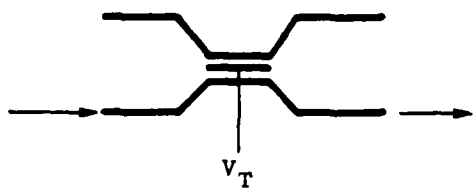
The switching is not very complete with the simple coupler switch shown in Figure 3.10. Different approaches have been proposed to achieve more complete switchings. One technique utilizes the directional couplers to construct a balance bridge as illustrated in Figure 3.11. The refractive index of one of the branches is varied by the application of a voltage. The recombined



$$V = 0 \begin{cases} I_{out\ 1} = I_{in} \\ I_{out\ 2} = 0 \end{cases}$$

$$V = V_T \begin{cases} I_{out\ 1} = 0 \\ I_{out\ 2} = I_{in} \end{cases}$$

DIRECTIONAL COUPLER WAVE GUIDE SWITCH



SCHEMATIC REPRESENTATION

FIGURE 3.10 DIRECTIONAL COUPLER WAVEGUIDE SWITCH.

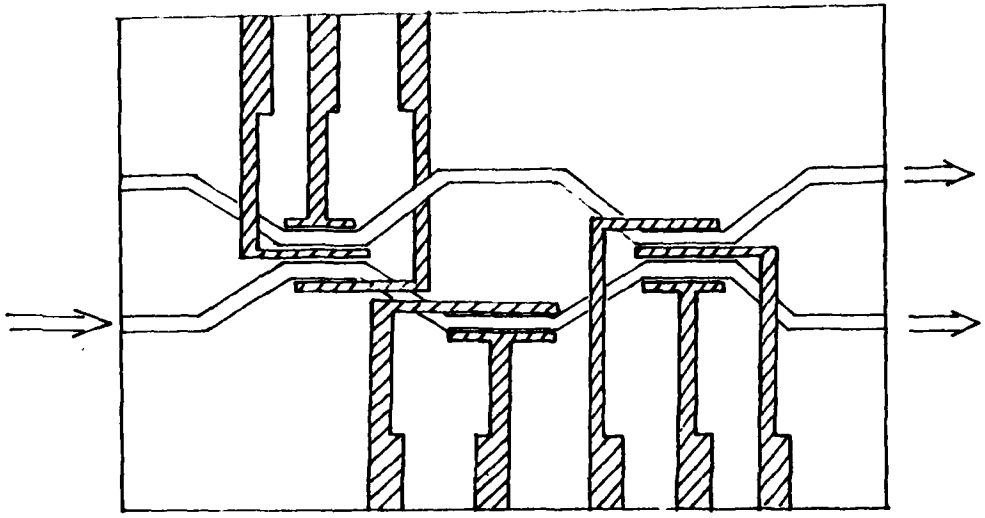


FIGURE 3.11 BALANCED BRIDGE ARRANGEMENT FOR DIRECTIONAL COUPLER OPTICAL SWITCH.

beams interfer either constructively or destructively to produce the switching effect[19]. Another technique makes use of an alternating $\Delta\beta$ structure as shown in Figure 3.12 to produce more complete switching[20]. This unique structure allows better control over the coupling and requires a substantially lower electrode voltage to achieve switching. Cross talk as low as -26 dB has been demonstrated.

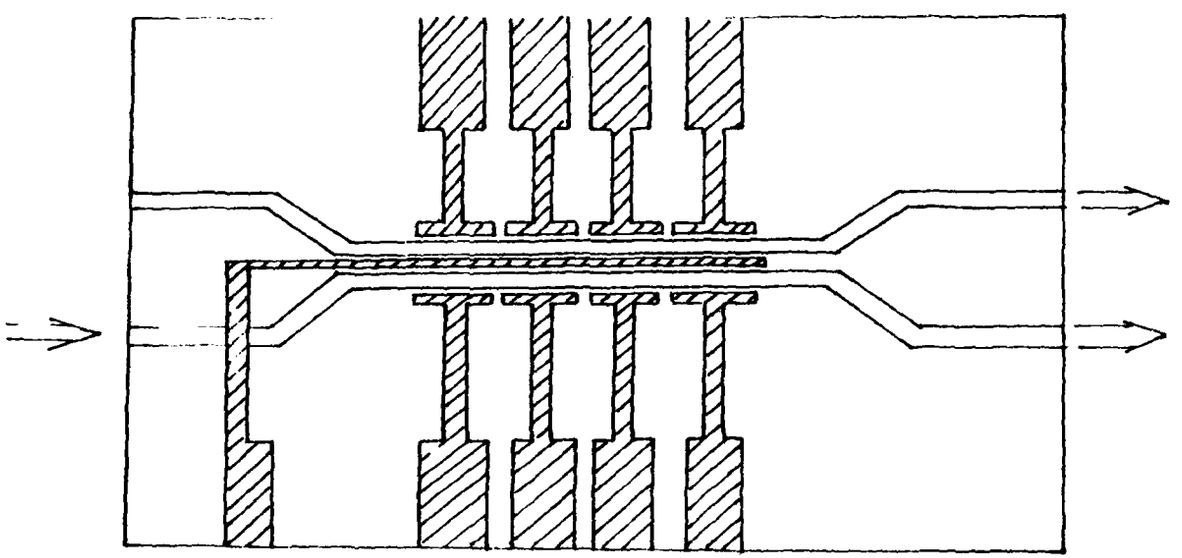


FIGURE 3.12 ALTERNATE ARRANGEMENT FOR DIRECTIONAL COUPLER OPTICAL SWITCH.

4
IMPLEMENTATION OF THE MAPPING APPROACH

4.1 INTRODUCTION

We have introduced in the last section several implementation concepts for optical residue computations. The numeric representations and computation mechanisms of these concepts are fundamentally different. In order to develop a definite design concept and evaluate its performance, it is necessary to base the design on an explicit set of hardware and specifications. To this end, we have chosen the mapping approach for our conceptual development. As pointed in the last chapter, the use of cyclic devices for numerical operations entails the use of analog devices for digital representation. This would involve quantization errors that could accumulate to an unacceptable level in long sequential operations. The spatial representation of the mapping approach is inherently binary; the advantage of low probability of error of a digital system is thus preserved. To reduce the quantization error with the cyclic implementation, a cyclic device with a multistable states characteristic would be necessary. While feed back devices that exhibit multistable states behavior have been demonstrated, none can yet produce a large enough dynamic range for the representation of large moduli. Furthermore, while sequential addition operation can be performed efficiently with the cyclic approach, the implementation of multiplication and fixed transformation is much more complex than the mapping approach. We should emphasize however, that our choice of mapping approach should not be taken as a definitive endorsement. While we believe that the mapping approach is promising and practical with the hardware available today, the cyclic approach also possesses unique advantages and its potential cannot be ignored. In the end, the most efficient approach may well be a hybrid combination of the features offered by both the mapping and cyclic approaches. In the following sections, we shall develop a design concept based on the mapping approach, using integrated

optics components of demonstrated capabilities. While the design described is very specific as to hardware implementation, the design concept is flexible enough to be adapted for different and better hardware that may be introduced in the future.

In the last section, we have discussed some spatial switching devices for guided and unguided light beam. They are all candidates for an optical residue computer using the spatial mapping approach. However, to be integrated into a small package, the devices that deflect unguided light beam (e.g., acoustical modulators) seem the least feasible due to the limits imposed by deflection angle and light diffraction. Among the devices that switch the light path of guided light beams, the directional waveguide coupler is the most promising at this time. The directional coupler is one of the better developed integrated optics devices due to its importance as a spatial multiplexer in fiber optics communication systems. The 2 input-2 output port feature of the coupler also offers maximum flexibility in the integrated optics circuit design. To demonstrate the design concept, we have therefore chosen the use of directional couplers for the implementation of the optical residue computer.

We have briefly described the mechanism of the directional coupler switch. To provide a better insight as to the physical size of the devices, we have illustrated in Figure 4.1, the typical dimension of a simple coupler switch. At present the coupler has a minimum length of about 3 mm, fairly large as compared to integrated electronic devices. However, the width of the coupler switch is only 20 μm. Thus many switches can still be packed into a relatively small area.

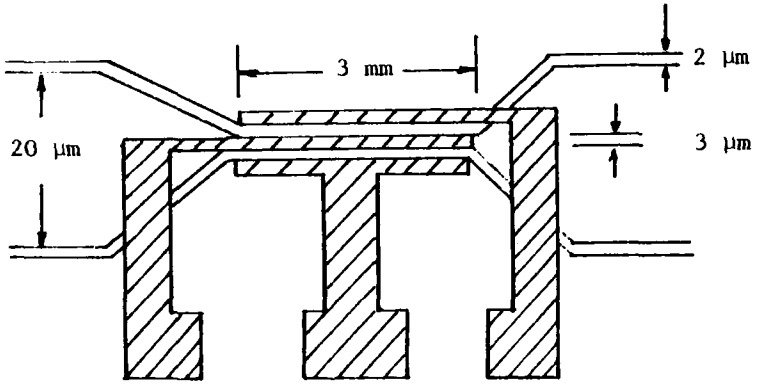


FIGURE 4.1 TYPICAL DIMENSION OF DIRECTIONAL WAVEGUIDE COUPLER.

4.2 IMPLEMENTATION OF ADDER, SUBTRACTER AND MULTIPLIER

The most basic arithmetic operator is the adder[21,22]. Addition in residue arithmetic is essentially a shifting operation. The input light beam is shifted by K positions for the operation +K as illustrated in Figure 4.2 for modulus 5. It is possible to construct a residue computer entirely out of such fixed maps. However, the storage and selection of a large number of hard wired maps would not be practical. The programmable map approach is in general more desirable. One possible implementation of a modulo 5 adder is shown in Figure 4.3. With this design, the electrode voltages of all the coupler waveguide switches are initially set at V_T. The light wave injected into the input of the adder will therefore propagate inside the same waveguide through the adder. To program the device for the +2 operation for example, the electrode voltage of the corresponding row of couplers is change to 0. Thus, when the light propagation reaches that particular set of coupler waveguide switches the light wave will be coupled into the adjacent waveguide, changing the optical path. The electrode voltages are maintained at constant levels of V_T or 0 by connecting the electrodes to a set of S-R flip flops. The adder can be programmed by sending an electric pulse to the 'S' input of the appropriate flip flop, triggering it to change state. Alternatively, we could let the initial electrode voltage of all the couplers be 0 and program the adder by changing the electrode voltage of a particular row of coupler switches to V_T. The alternate design of a modulo 5 adder is shown in Figure 4.4. However, we generally find that it is easier to trace the light path with the

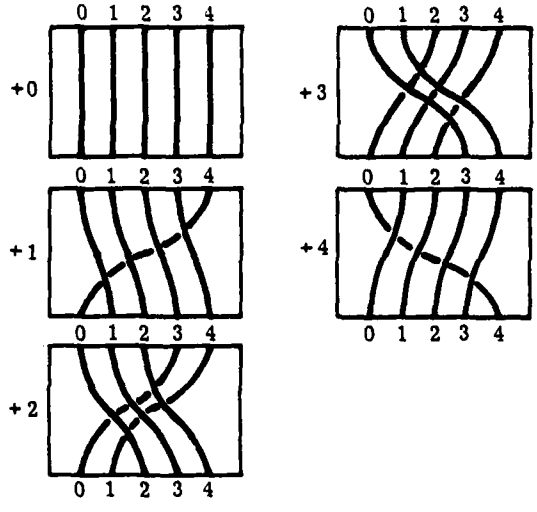


FIGURE 4.2 FIXED MAPS FOR MODULO 5 ADDITION.

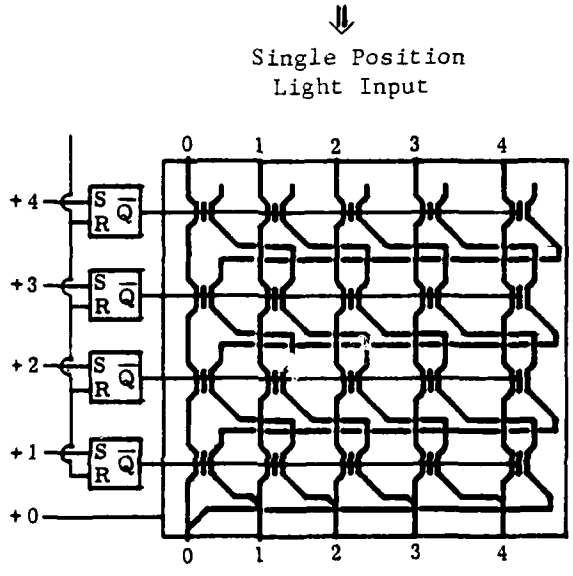


FIGURE 4.3 IMPLEMENTATION OF MODULO 5 ADDER

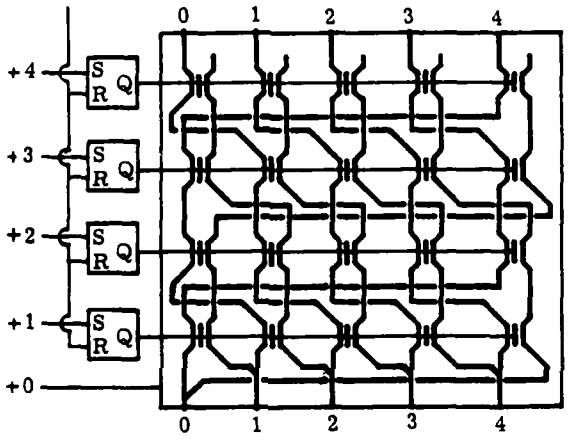


FIGURE 4.4 ALTERNATE DESIGN FOR MODULO 5 ADDER.

former design and to make the devices easier to study, we shall make use of the former design in this report. We shall also use the term 'on' to describe the state where coupling occurs at the coupler switch and term 'off' for the state where the light propagation will remain in the same waveguide.

Subtraction can be performed with the use of the additive inverse. The additive inverse $|-K|_{m_i}$ of a residue number K is defined such that

$$\left| K + |-K|_{m_i} \right|_{m_i} = 0 \qquad (4.1)$$

There is a fixed one-to-one correspondence between a residue number and its additive inverse. The additive inverse transformation can therefore be implemented by a fixed map. And by adding this transformation map to an adder, one can convert it into a subtractor as shown in Figure 4.5 for modulus 5.

Multiplication can be implemented directly by using m_i maps for the operations of x0, x1, x2, ..., $xm_i - 1$. Alternatively, one can make use of a homomorphic approach where a modulus m_i multiplication is converted into a modulo $m_i - 1$ additive operation[2]. A $\log_b K$-like forward transform is first performed on the operands. A modulo $m_i - 1$ addition is then performed and the sum is inverse transformed by a b^K-like transform to obtain the product of the two original numbers. The transform table for modulus 5 is shown in Figure 4.6(a), and the process is illustrated schematically in Figure 4.6(b). Although the $\log_b K$-like transformation for the value 0 is not defined, it is known that if either the multiplier or the multiplicand is 0, the product is 0. A modulo 5 multiplier is shown in Figure 4.7 using this homomorphic approach. We note that for a modulo 5 multiplication, a

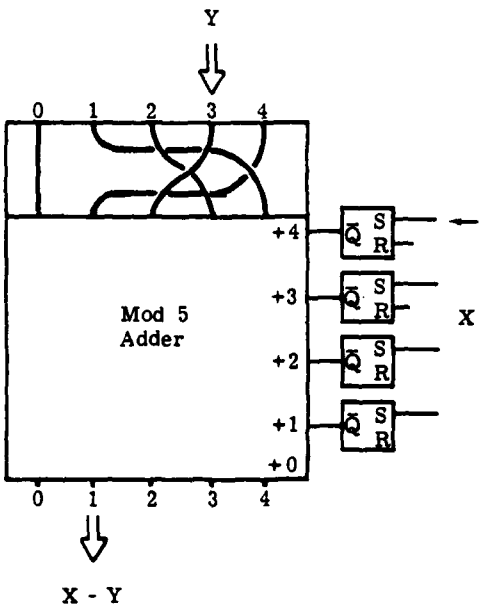


FIGURE 4.5 CONVERTING AN ADDER FOR SUBTRACTION OPERATION.

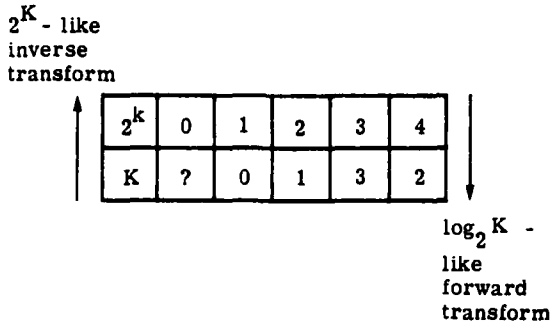


FIGURE 4.6A TRANSFORM TABLE FOR MODULUS 5.

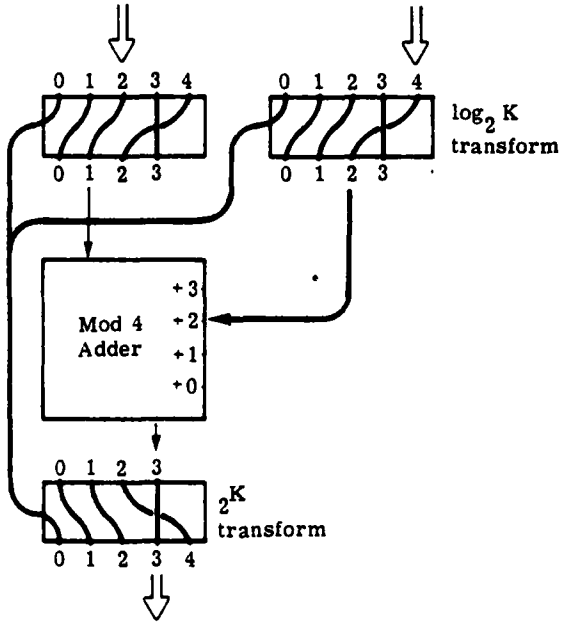


FIGURE 4.6B MODULO 5 MULTIPLICATION USING THE HOMOMORPHIC APPROACH.

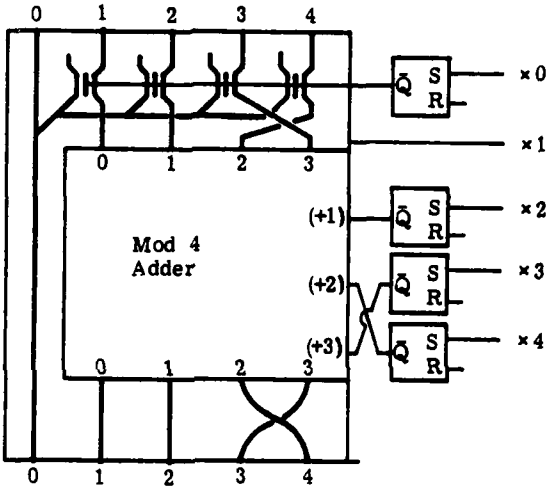


FIGURE 4.7 IMPLEMENTATION OF A MODULO 5 MULTIPLIER.

modulus 4 addition is performed. Thus, in order to convert a modulo 5 adder into a modulo 5 multiplier, the modulo 5 adder should be designed in such a way that it can be easily converted into a modulo 4 adder. This can be achieved with the design shown in Figure 4.8. While the concept can be applied to an adder of any modulus, we should note that this homomorphic approach can be used only if the modulus is prime.

The feature of this design is that the input, output and programming controls are all represented spatially in the same way. This allows the interconnection of these devices for sequential operations. The outputs of one module can be connected directly to the inputs of the next module or it can be used to program the map of the next adder as illustrated in Figure 4.9. An electrical pulse is sent to the first multiplier to program it to perform $x|X|_{m_i}$ where m_i is the modulus. A light pulse is then injected into the adder at the spatial position corresponding to $|Y|_{m_i}$. The exit position of the light beam would correspond to $|X \times Y|_{m_i}$. A fast avalanche photodiode is connected to each of the output wave guides. The existing light pulse will be detected by the photodiode, generating an electric pulse. The electric pulse in turn triggers the corresponding flip flop of the next adder, setting it for the $+|X \times Y|_{m_i}$ operation. Another light pulse is then injected into the input of the second adder at the position corresponding to Z_{m_i}. The position where the light pulse exits will represent the sum of $|X \times Y + Z|_{m_i}$

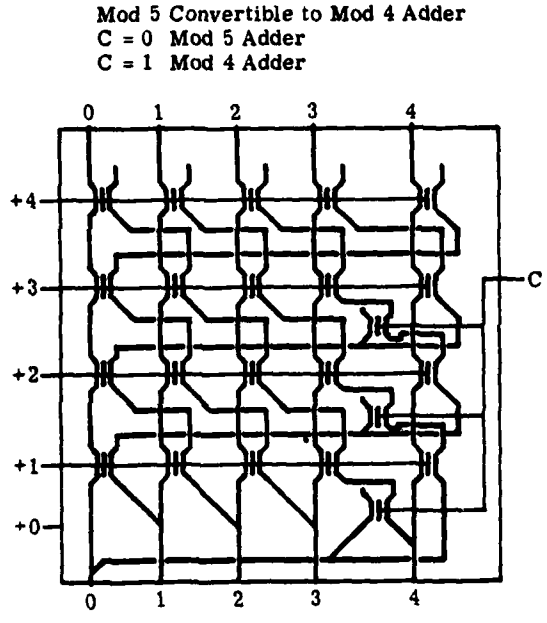


FIGURE 4.8 MODULO 5 ADDER CONVERTIBLE TO MODULO 4 ADDER.

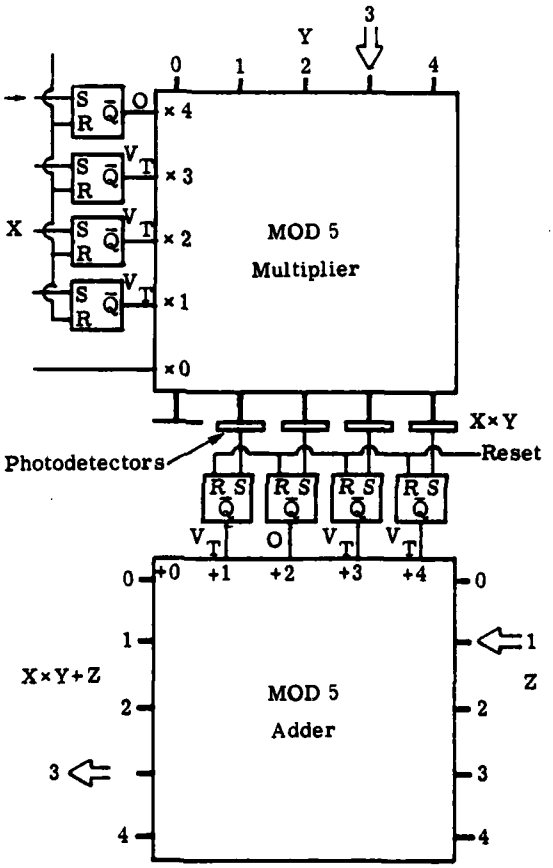


FIGURE 4.9 INTERCONNECTION OF MOD 5 MODULES.

4.3 PROGRAMMABLE MULTI-PURPOSE COMPUTATION MODULE

With the subunits described above, we can proceed to describe the multi-purpose programmable computation module. The module will contain four distinct parts as shown in Figure 4.10. Each of these subunits can be turned on and off individually, allowing the different combinations of the subunits to perform various computation operations. However, it is more complicated than simply stacking all the subunits together. Special attention must be paid to the case of +0 and x0 by noting that $X + 0$, $X \cdot 0 = 0$, $0 \cdot Y = 0$ and $X \cdot 1 = X$. Furthermore, the modulus m_i adder must be modified to perform modulus $m_i - 1$ addition and the $|-K|_{m_i}$ additive inverse transform must be converted into a $|-K|_{m_i-1}$ transform when the module is programmed to perform multiplication and division. A possible design of the programmable multipurpose computation module is shown in Figure 4.11.

The multi-purpose computation module can be programmed to perform +, -, × and ÷ arithmetic operations with simple binary controls. For example, to perform modulo 5 addition, the subunits for $\log_2 K$-like transform, additive inverse transform and 2^K-like transform are all turned 'off'. That is, light pulse injected into any of the m_i input ports will propagate undeviated along the same wave guide through these subunits. With these units 'off', the module would be essentially the simple adder shown earlier in Figure 4.3. To perform subtraction, the additive inverse transform $|-K|_{m_i}$ unit is turned 'on', changing the light path according to the transform map shown in Figure 4.6. We note that while operating in the addition and subtraction modes with the $\log_2 K$-like transform unit off, an input to the '*0' control has no effect on the light path. The position of the exit beam would therefore be the same as that of the input beam, performing in effect,

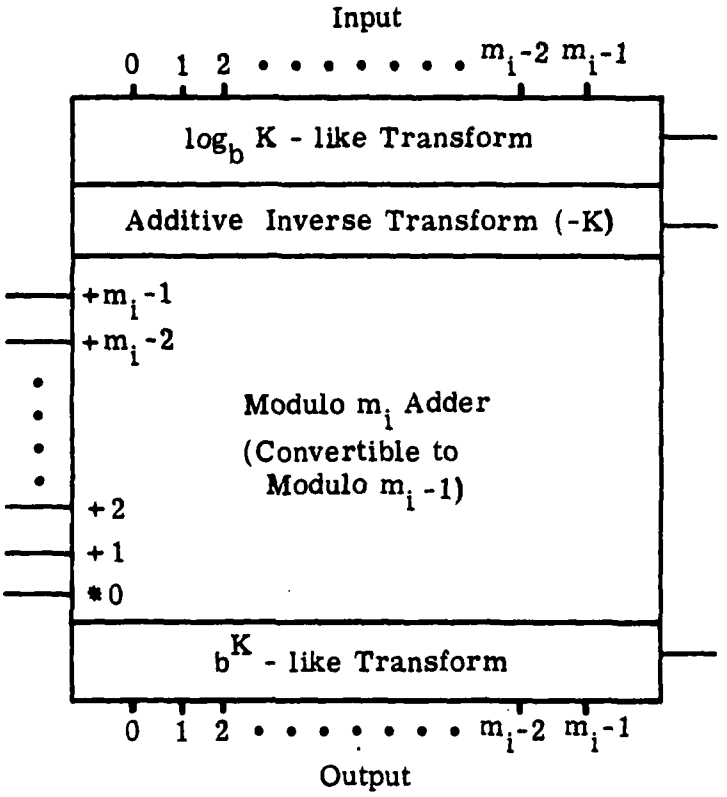


FIGURE 4.10 CONCEPTUAL DESIGN OF PROGRAMMABLE MULTI-PURPOSE COMPUTATION MODULE.

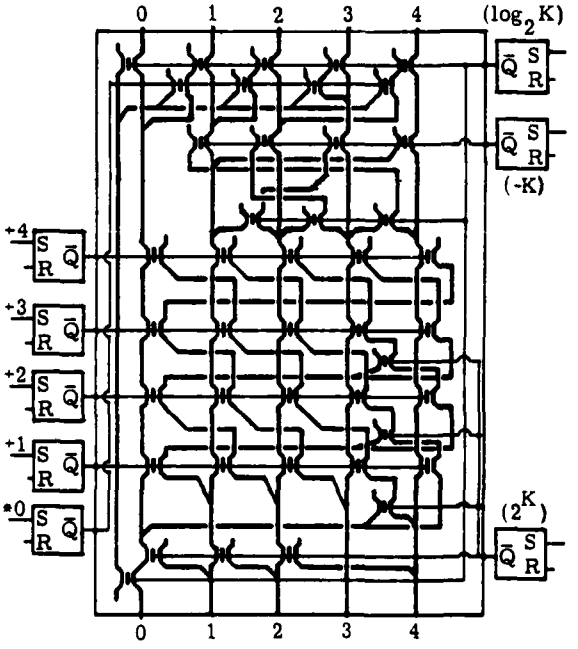
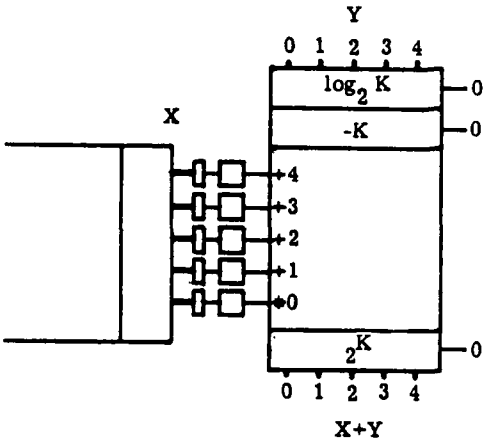


FIGURE 4.11 IMPLEMENTATION OF PROGRAMMABLE MULTI-PURPOSE COMPUTATION MODULE.

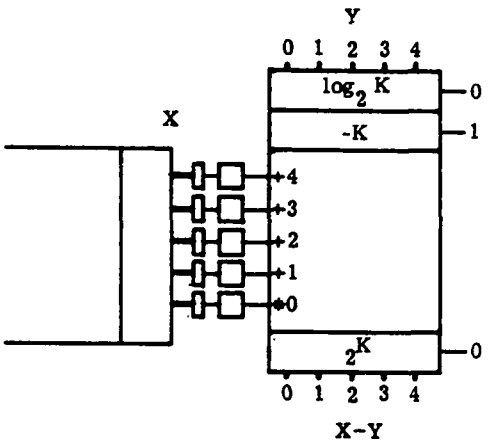
the +0 operation. The programming of the computation module for addition and subtraction operations are illustrated in Figures 4-12(a) and (b).

In programming the computation module for multiplication, there are two possible approaches. The module can be connected as a multiplier by rerouting the electrode leads to perform the $\log_2 K$-like transform on the multiplier value (X). The 2^K-like transform unit is turned on to inverse transform the sum as illustrated in Figure 4-12(c). With the second approach, both the multiplier (X) and the multiplicand (Y) values are transformed by computation modules, as illustrated in Figure 4.12(d). This approach has two advantages. The connection of the electrode leads do not have to be changed, allowing the module to be switched back to addition mode when desired. Secondly, it would provide more flexibility in performing division. Observe that the extra coupler switch at the left lower corner in Figure 4.11 is necessary for the module to be programmed in this mode. The coupler is turned on together with the $\log_2 K$-like transform unit at the top. When the value of the multiplier X is 0, the '*0' control of the second module is turned on, and the x0 operation is performed. If the multiplier is 1, its $\log_2 K$-like transform is 0; the purpose of the extra coupler switch is to keep the transformed output of the multiplier from setting the *0 control of the second module. Instead, the coupler switches the light path away from the 0 output port such that the second module would be left undisturbed. The light pulse will exit at the same position as it enters the module, performing the x1 operation.

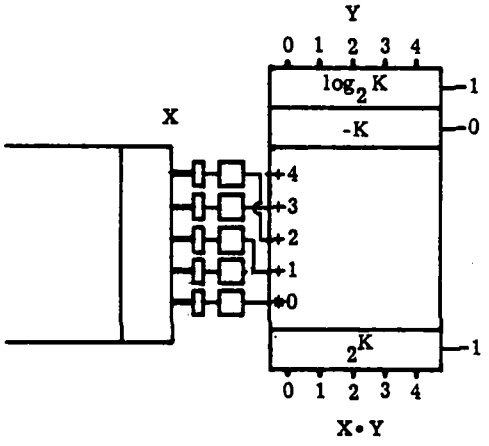
As pointed out in Section 2, division can be performed using the same homomorphic approach, converting a modulo m_i division into a modulo $m_i - 1$ subtraction if the quotient is an integer (i.e., no remainder). For reasons discussed before, residue arithmetic is generally applied to problems that do not require division operation such as matrix multiplication. Nevertheless, it would be useful to be able to perform division even if it is limited to the remainder zero case. One operation that requires such division operation



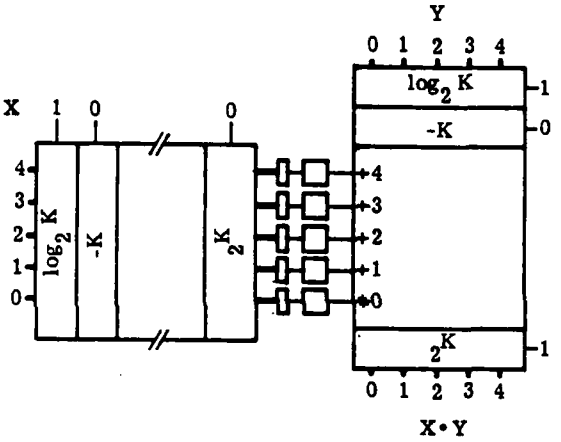
(a) For addition



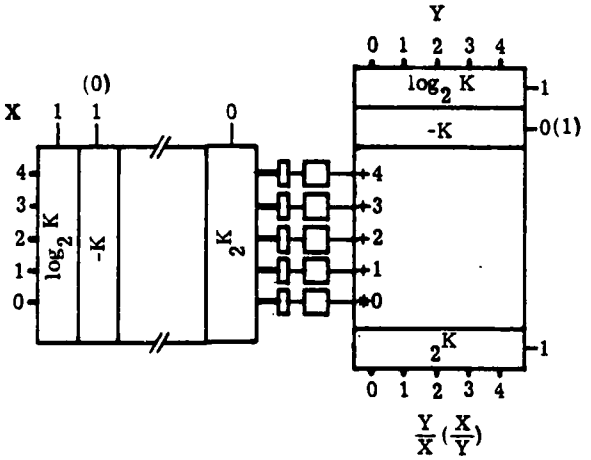
(b) for subtraction



(c) for multiplication with electrode leads rerouted



(d) for multiplication with alternate arrangement



(e) for division

FIGURE 4.12 PROGRAMMING OF COMPUTATION MODULE.

is scaling. In order to keep the values within the range of the residue number system, it may be necessary to periodically scale the values down by a factor of K. We have shown that scaling can be achieved by division if K is a value of one of the moduli or the product of two or more moduli. The programming of the computation module for the division operation is illustrated in Figure 4.12(e). An $|-K|_{m_i-1}$ additive inverse transform is required for the divisor after the $\log_2 K$-like transform. A $|-K|_{m_i}$ transform can be changed into a $|-K|_{m_i-1}$ transform by shifting down the values of the $|-K|_{m_i}$ transform by 1. Referring back to the module design shown in Figure 4.11, the down shifting is performed by the set of three switches at the fourth row. They are turned on together with the $\log_2 K$-transform unit.

4.4 MATHEMATIC OPERATIONS

We have demonstrated the uses of the computation modules for various basic arithmetic operations such as multiplication and addition. In the next few sections, we shall apply the computation modules to more complicated mathematical operations such as polynominal evaluations, matrix multiplications, correlations and Fourier transformations. These operations are quite representative of those often encountered in signal processing. They have one common feature; there is no general division required in the computation. In the implementation, parallel structures are used whenever possible to optimize speed and pipelining is used to maintain a high throughput rate. With such a parallel processing approach we shall show that it is possible to achieve a throughput rate over 300 MHz for these computations.

4.4.1 EVALUATION OF POLYNOMIALS

To demonstrate how the computation module can be interconnected to perform various mathematical calculations, we first apply it to the evaluation of polynomials. As discussed by Huang et. al, a polynomial may be evaluated using a single map. For example, the modulo 5 map for the computation of $X^3 + 4X^2 + 3X + 2$ is shown in Figure 4.13. However, to generate that map, one would require the help of some external intelligence. The routings of the m_i possible inputs have to be computed beforehand. This implementation is therefore not easily programmable. An alternative is to utilize a set of fixed maps for X^n, X^{n-1}, ... X^2 functions in conjunction with the computation modules as shown in Figure 4.14. To perform the modules for the computation of $X^3 + 4X^2 + 3X + 2$ for example, the coefficients 1, 4, and 3 are entered into the multipliers. Light pulses are injected into the inputs of the multiplier at the ports corresponding to the value of input X. The adders would be set by the output of the multipliers for $+(X^3)$, $+(4X^2)$ and $+(3X)$ operations. Another light pulse is then entered into the first adder at input port 2, and the position where the light pulse exits would correspond to the value of $X^3 + 4X^2 + 3X + 2$.

The computation time would be equal to the time needed to set the adder module plus the propagation time through four modules. The propagation time through a single module of 1/2 inch size would be about 40 psec. The set time of the module is the sum of the detection delay of the photodiode, the switching delay of the flip flop and the switching time of the waveguide coupler. It is possible to achieve a set time under 2 nsec for the computation module[23,24]. And if we assume that an additional lnsec is required for the light pulse to pass through the module and to reset the flip

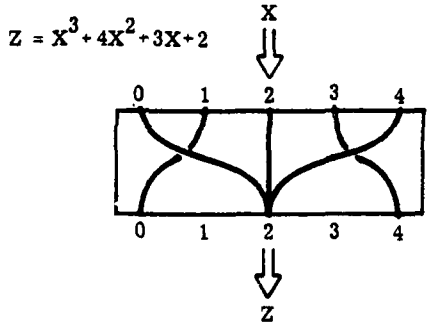
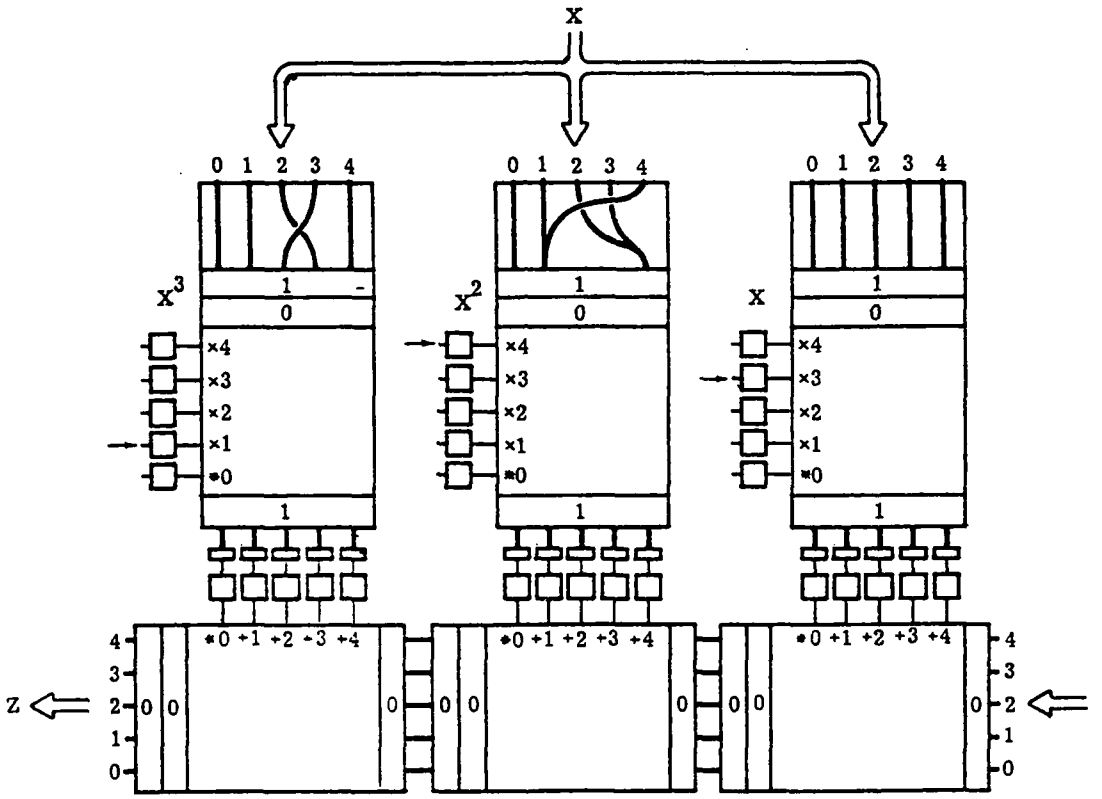


FIGURE 4.13 EVALUATION OF A POLYNOMIAL WITH A SINGLE MAP.



$$Z = (X^3 + 4X^2 + 3X + 2); \quad X=2$$

FIGURE 4.14 PROGRAMMABLE ARRANGEMENT FOR THE EVALUATION OF POLYNOMIALS.

flops, the set-reset cycle time for the computation module would be about 3 nsec. The throughput rate for the evaluation of third order polynominal would then be

$$\frac{1}{3.12 \text{ nsec}} = 320 \text{ MHz}$$

Due to the parallelism of the arrangement, the computation time is approximately the same for polynominals of any order.

4.4.2 MATRIX MULTIPLICATION

One of the important applications of the numerical optical computer is the multiplication of matrices. It can be extended to a number of transform operations such as DFT, Hadamard transform, etc. We shall examine the general case of matrix multiplication, $[A]_{MxN}\ [B]_{NxP} = [C]_{MxP}$. The coefficients of the master matrix $[B]_{NxP}$ are stored in the modules as multipliers as shown in Figure 4.15. The values of the matrix $[A]_{MxN}$ pass through the multipliers row by row setting the corresponding row of adders. Light pulses are entered into the first adder of each row, providing in parallel the values of the first row of $[C_{1j}]$ at the output. The flip flops are then reset, ready for the entries of the next row of $[A]_{MxN}$. The total computation time is equal to M + 1 set-reset times of the module and P + 1 propagation time. The number of computation modules required is 2NP. For example, to multiply two 10 x 10 matrices, the computation time would be about 34 nsec if we assume a module set-reset time of 2 nsec and the use of 200 computation modules for each modulus. The coefficients of $[A]_{MxN}$ could be for example, the encoded signals from an array of sensors. The signals are sampled and entered into the optical residue row by row at the system throughput rate.

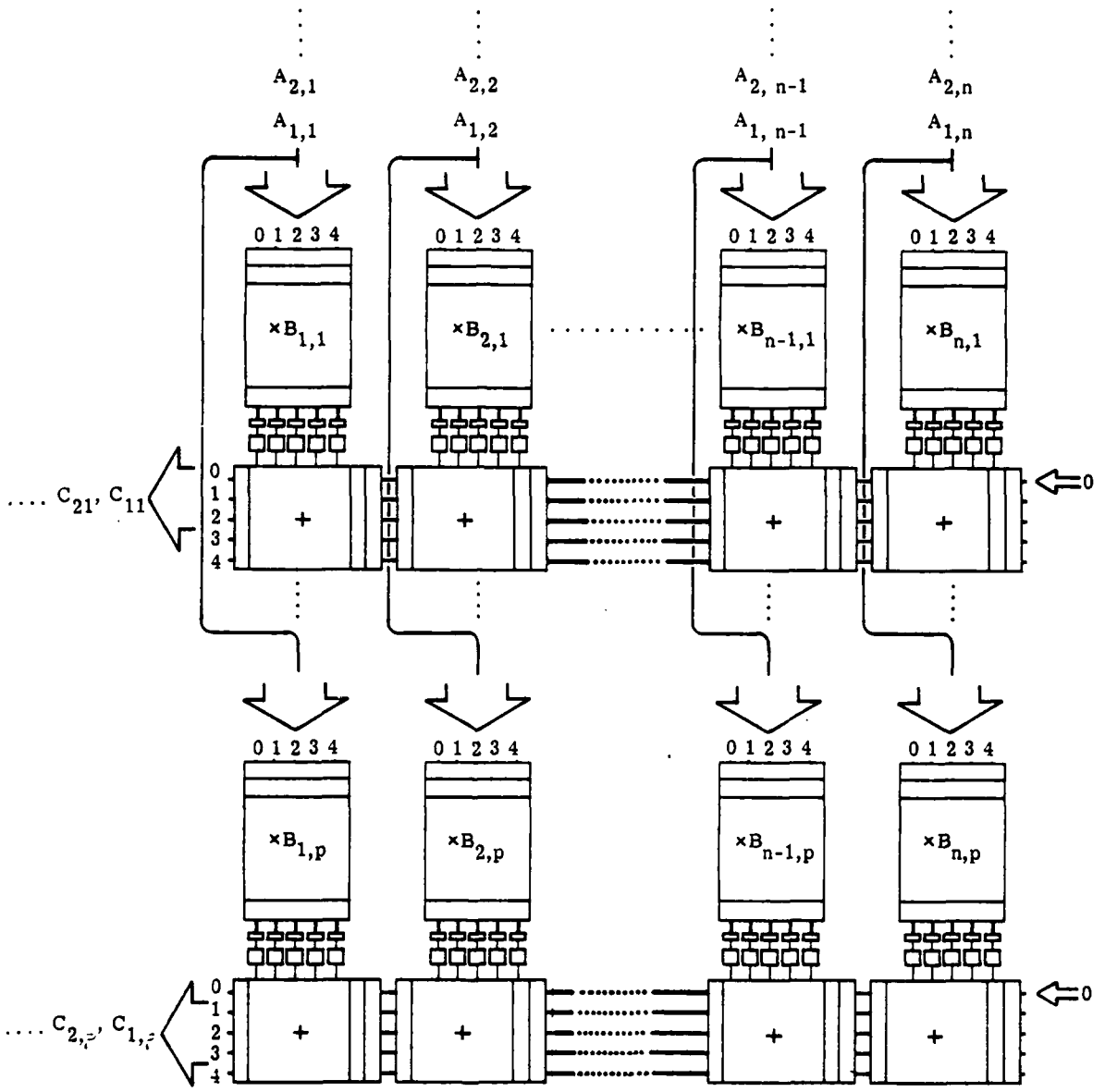


FIGURE 4.15 MATRIX MULTIPLICATION.

4.4.3 CONVOLUTION AND CORRELATION

In performing matrix multiplication $[A]\cdot[B] = [C]$, we have

$$C_{ij} = \sum_{k=1}^{n} a_{ik}\, b_{kj} \tag{4.2}$$

Two special cases of matrix multiplication that are of special interest in signal processing are

$$C_t = \sum_{n=1}^{N} b_n\, a_{t-n} \tag{4.3}$$

and

$$C_t = \sum_{n=1}^{N} b_n a_{t+n} \tag{4.4}$$

They correspond to the convolution and correlation operations respectively.

In Figure 4.16, the conceptual implementation of the convolution operation is illustrated. With this scheme, the reference function B_n is stored in the form of multipliers and the input data A_{t-n} are propagated through the parallel multipliers at a rate of $1/\tau$ where τ is the cycle time for one complete sum of product operation. The

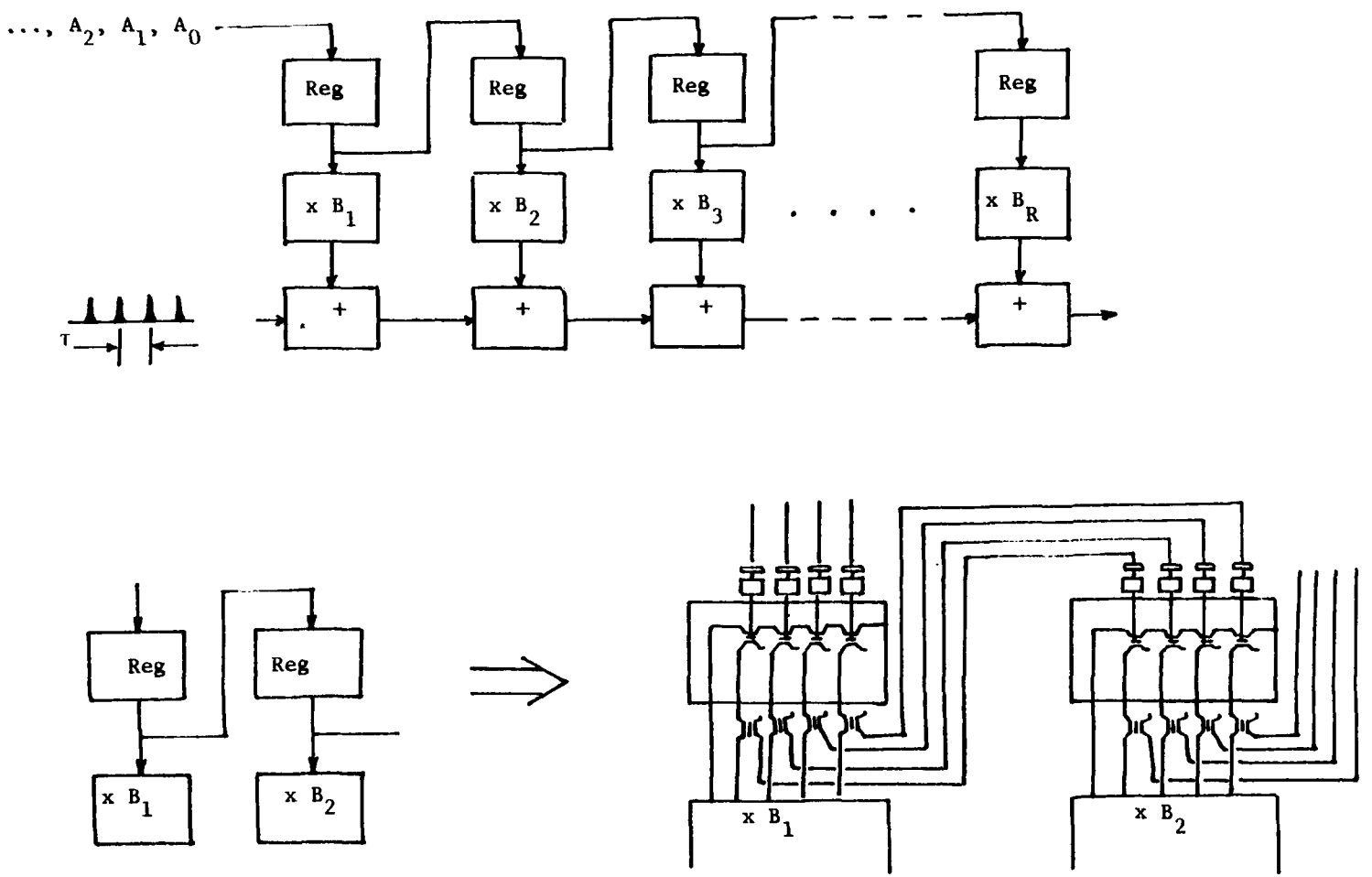


FIGURE 4.16 ARRANGEMENT FOR CORRELATION OPERATION.

input data are shifted from one multiplier to the next with a series of optical data registers. Alternatively, the multipliers can be set by the input data through a series of electronic shift registers as illustrated in Figure 4.1.7. With this design, the reference function is represented by the positions where light pulses are injected into the multipliers and the input data a_{t-n} are used to program the multipliers. After each cycle of computation, input data are shifted down one multiplier, reprogramming them for the next set of computation. The computation speed of the two schemes are about equal. However, the second scheme may be simpler to implement.

It may be interesting to note that with an electronic computer, the convolution and correlation operations can be performed more efficiently using the transform techniques than with the direct computation of Eq. 4.3 and Eq. 4.4. Multiplication is the most time consuming computation operation in an electronic digital computer. The computation time required for an algorithm is therefore determined largely by the number of sequential multiplication steps. With the use of the Fast Fourier Transform (FFT) technique, convolution and correlation can be performed with less multiplication operations than the direct computation of Eq. 4.3 and Eq. 4.4. The same however, is not true for the optical residue computer where multiplications can be performed at essentially the light propagation speed. The computation speed is determined instead, by the set-reset cycle time of the computation modules and the number of sequential steps.

4.5 PIPELINING CONCEPT

With the systems described above, the light pulse has to propagate through a long series of adders. There are two disadvantages associated with such a scheme. First of all, the

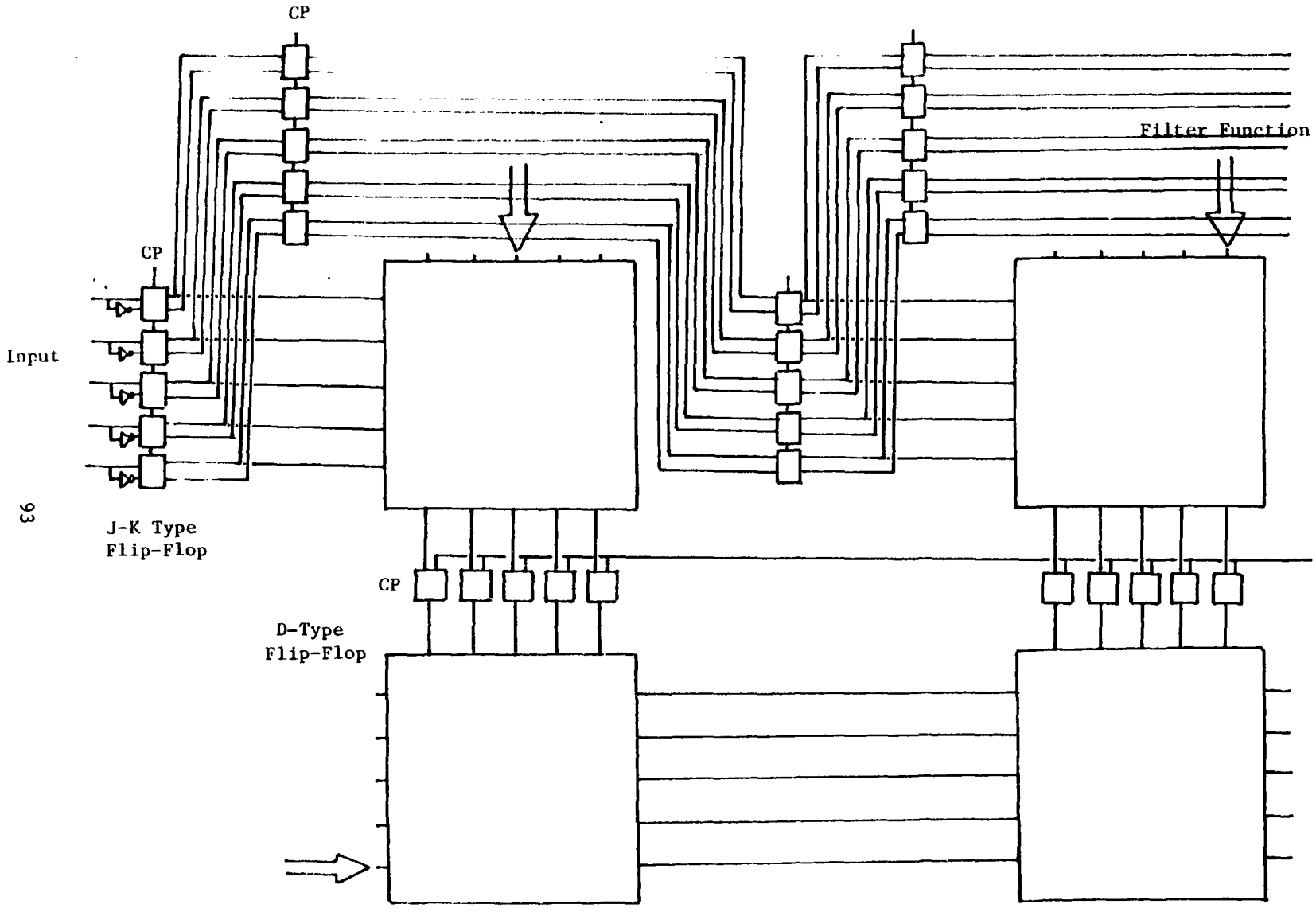


FIGURE 4.17 CORRELATION COMPUTATION WITH THE USE OF ELECTRONIC SHIFT REGISTERS.

optical loss through a module is quite high. Passing through a large number of modules, the intensity of the light pulse could be reduced to an undetectable level. Secondly, the cycle time per computation is increased. If the number of modules that light pulses have to propagate through is small, then the propagation delay may not be as significant a factor on the computation speed as the set-reset time of the computation module. However, if the number of modules is large, the propagation time could be the determining factor for computation speed of the system. The number of modules that a light pulse has to go through should therefore be kept small to achieve optimal computation speed and bit error rate.

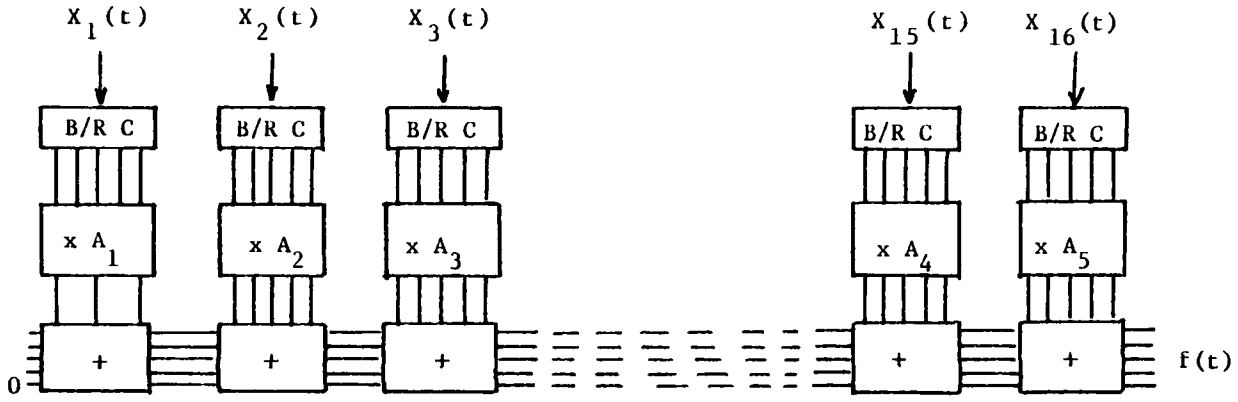
As an illustration, we shall look at the operation

$$\sum_{i=1}^{N} a_i \, x_i$$

To obtain a quantitative comparison, let us assume that N = 16. The basic system structure is shown in Figure 4.18a. If we assume a module set-reset time of 3 nsec and propagation time of 40 psec, the throughput rate would be

$$\frac{1}{3 + 17(.04)\text{nsec}} = 271.7 \text{ MHz}$$

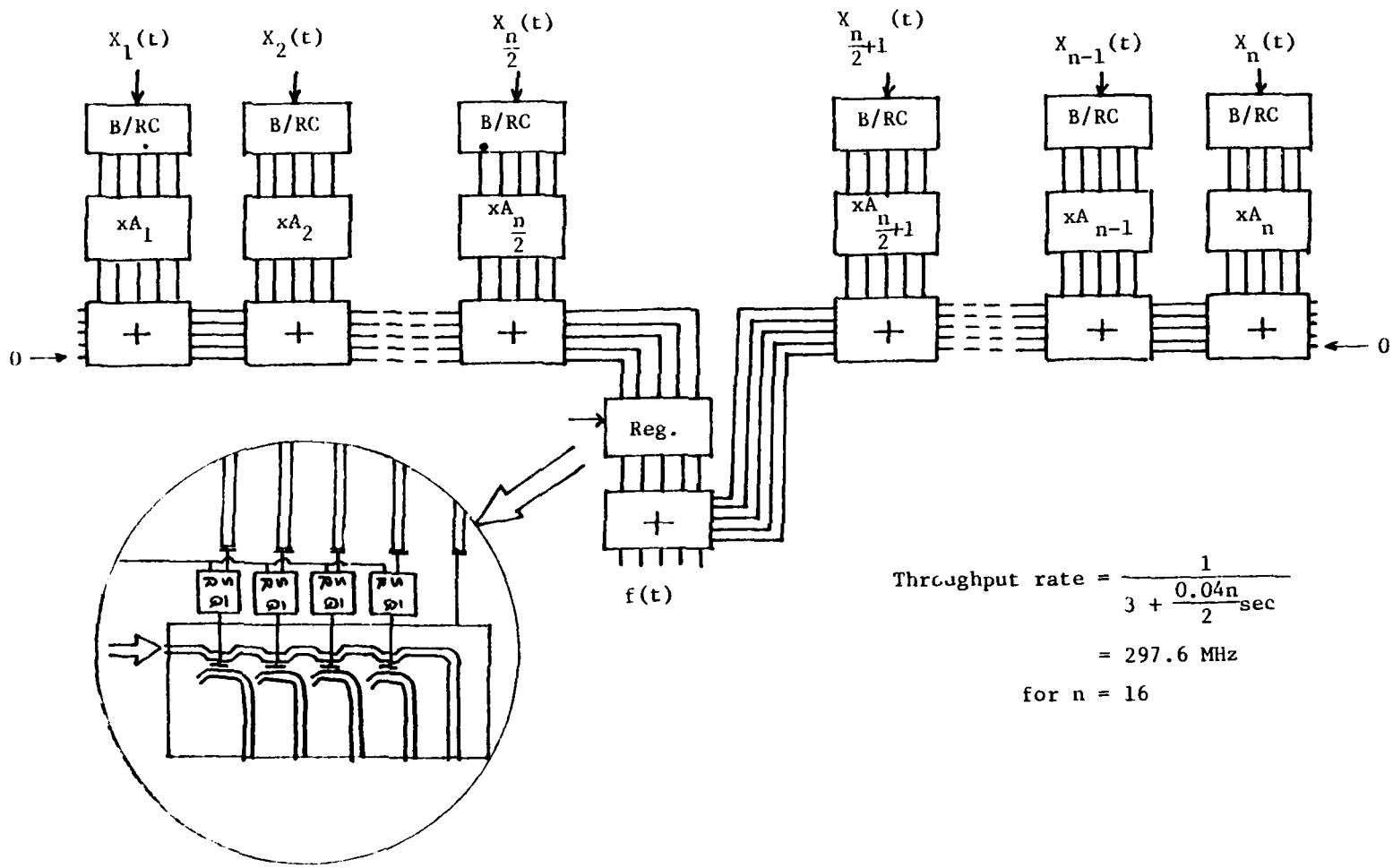
The initial delay in obtaining the first output is about 3.69 nsec and the light pulse has to propagate through 16 consecutive modules. In Figure 4.18b we show an alternate arrangement where the light pulse needs only to propagate through 8 modules. It requires an additional step but the throughput rate can be maintained by pipelining the operation. The output of one side of the summation is used to program an adder while the sum of the other half is stored in a



$$\text{Throughpyt rate} = \frac{1}{3.68 \text{ nsec}} = 271.7 \text{ MHz}$$

Initial delay = 2.5 nsec.

FIGURE 4.18A PROCESSOR STRUCTURE WITHOUT PIPELINING.



$$\text{Throughput rate} = \frac{1}{3 + \frac{0.04n}{2}\text{sec}}$$

$$= 297.6 \text{ MHz}$$

for n = 16

FIGURE 4.18B FIRST LEVEL OF PIPELINING.

register. At the next clock cycle, the data is recalled from the data register and propagated through the adder, giving the value of the full sum. At the same time, the next set of input data X_{i+1} will go through the multipliers and reprogram the adder. Since the light pulse at each cycle has to go through a maximum of only eight modules, the throughput is now increased to

$$\frac{1}{3 + 8(.04)\text{nsec}} = 297.6 \text{ MHz}$$

Although the 6.4 nsec initial delay is longer, for a long series of continuous computations, the increase in throughput rate would more than make up for the lengthened initial delay. If we carry the concept to the farthest, we obtain an arrangement as shown in Figure 4.19. Now the light pulse will only have to propagate through a maximum of 2 modules and the throughput rate is increased to 324.7 MHz. The initial delay would also be increased to 12.3 nsec.

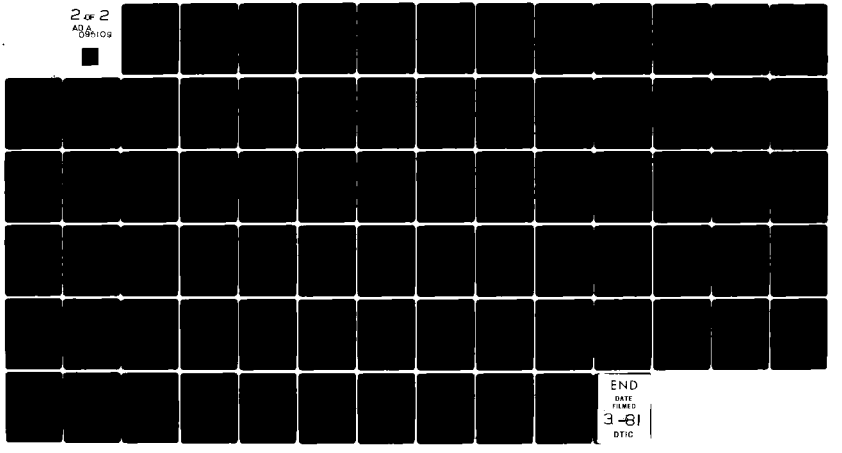
If the number of input data is small, then the increase in throughput rate cannot offset the additional initial time delay. The light pulse should then be propagated through as many modules as allowable by the optical loss. If the series of input data is very long, then the highly cascaded version of Figure 4.19 should be used since the increase in throughput rate becomes a significant factor. In addition, the probability of error would be lowered due to reduced optical loss.

4.6 DISCRETE FOURIER TRANSFORM

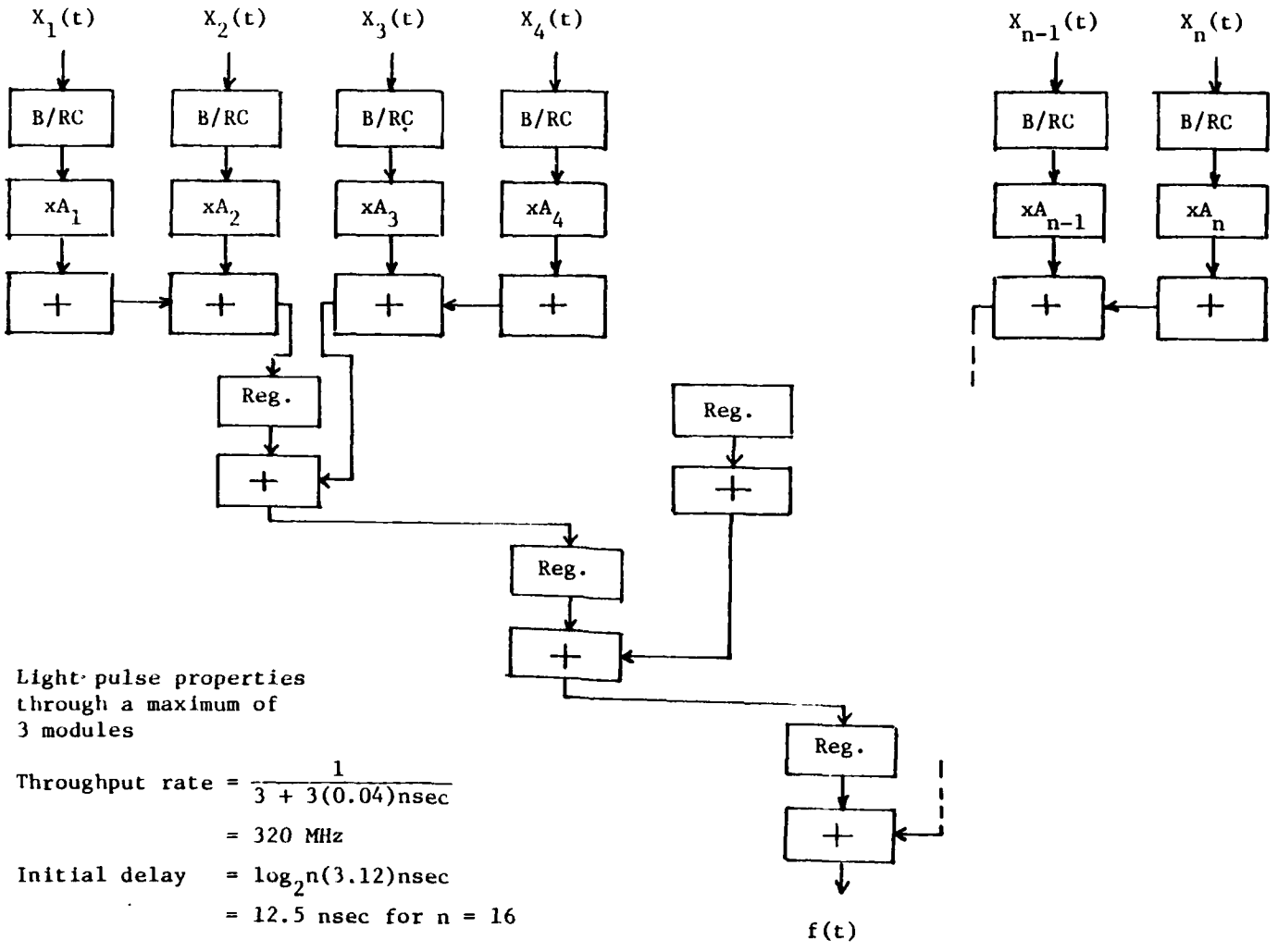
One of the most powerful transform operations for signal processing is the Fourier transformation. Optical Fourier transformation with coherent light and lens system is the heart of coherent optical processing while the FFT computation is the fundamental operation in digital signal processing.

AD-A095 109 ENVIRONMENTAL RESEARCH INST OF MICHIGAN ANN ARBOR RA--ETC F/G 20/6
DESIGN CONCEPTS FOR AN OPTICAL NUMERICAL COMPUTER BASED ON THE --ETC(U)
JUN 79 A M TAI, I CINDRICH, J R FIENUP DAS660-78-C-1035
UNCLASSIFIED ERIM-136800-14-F NL

2 OF 2
AD A095109



END
DATE FILMED
3-81
DTIC



Light pulse properties through a maximum of 3 modules

Throughput rate $= \frac{1}{3 + 3(0.04)\text{nsec}}$

$= 320$ MHz

Initial delay $= \log_2 n(3.12)\text{nsec}$

$= 12.5$ nsec for $n = 16$

FIGURE 4.19 FULLY PIPELINED PROCESSOR STRUCTURE.

4.6.1 DFT

The discrete Fourier transformation (DFT) can be expressed as

$$f(k\Omega) = \sum_{n=0}^{N-1} f(nT)e^{inTk\Omega}$$

$$= \sum_{n=0}^{N-1} f(nT)e^{i\frac{2\pi}{N}n} \qquad (4.5)$$

where

$$\Omega = \frac{2\pi}{NT}$$

f(nT) is generally a function of time or space and T is the temporal or spatial increment between samples. Since $e^{inTk\Omega}$ is complex, it can be expressed as a + ib. However, a and b are less than or equal to 1 and fractional numbers can not be represented by the residue number system. The values of a and b must therefore be scaled up and rounded off into integers. That is

$$2^M e^{inTk\Omega} \cong A + iB$$

To obtain the correct values of $f(k\Omega)$, the output must be rescaled by 2^{-M}. This may be accomplished more conveniently after the outputs of the residue computers are decoded into binary form. The down scaling can be achieved by simply shifting the output values down by M binary digits.

Let us first consider the case where f(nT) is real. This would correspond to applications where the inputs are signal intensities. For this special case, the real and imaginary part can be operated on separately and independently by noting that (A + iB) C = AC + iBC. One possible scheme is to store the input values f(nT) in the form of

N multipliers as illustrated in Figure 4.20. The scaled and round-off values of the coefficients $e^{inTk\Omega}$ are then recalled from a ROM and entered into the multipliers. The ROM can be a conventional electronic unit and the coefficients are stored in the residue form (e.g., 001000 for 2 mod 7). The binary digits are recalled in parallel and used to set the flip flops of an optical data register which is connected to the inputs of the multiplier as shown in Figure 4.21. The throughput rate of such a system would be $\frac{1}{t\text{nsec}}$ where t is the access time of the electronic ROM. While the access time of state of art ROM can be as short as 20 nsec, it is still substantially longer than the cycle time of the optical computation module. Alternatively, a series of optical data registers that are shifted cyclically can be used to maintain the throughput rate of the optical residue computer. The arrangement is illustrated in Figure 4.22. Another possibility is the use of holographic memories as shown in Figure 4.23. We note the total number of spatial positions for all moduli is only

$$\sum_{i=1}^{N} m_i$$

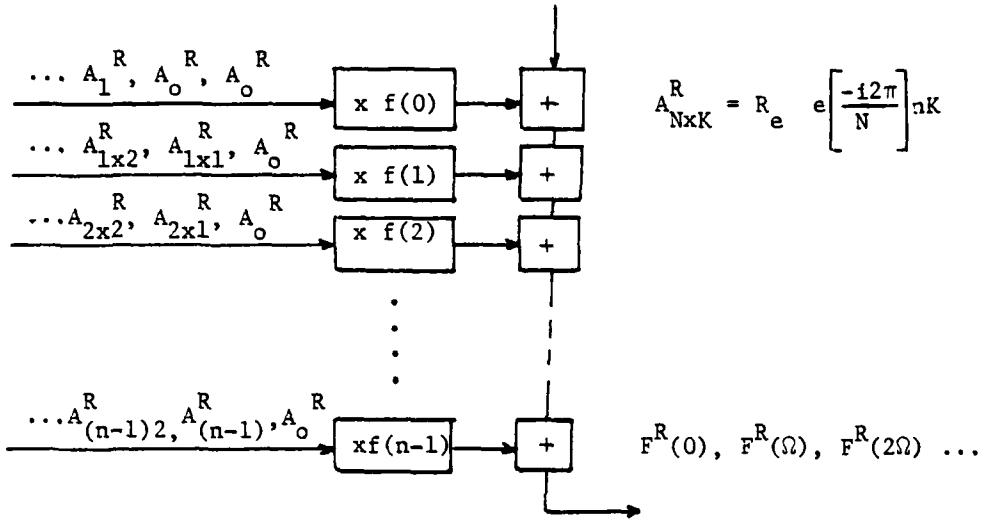
and the recorded data of a coefficient for all the moduli can be stored in the same sub-hologram. The data are reconstructed together and focused onto a bundle of optical fibers which lead the light pulse to the input ports of the multipliers. The read beam is scanned across the holographic memory, reading out sequentially the appropriate coefficients.

In the discussion above, the inputs f(nT) are assumed to be real. However, the values of f(nT) are in general complex. For the multiplication between two complex numbers, the real and imaginary parts can no longer be treated separately. Complex multiplication

D.F.T. SIMPLEST CASE: INPUT REAL

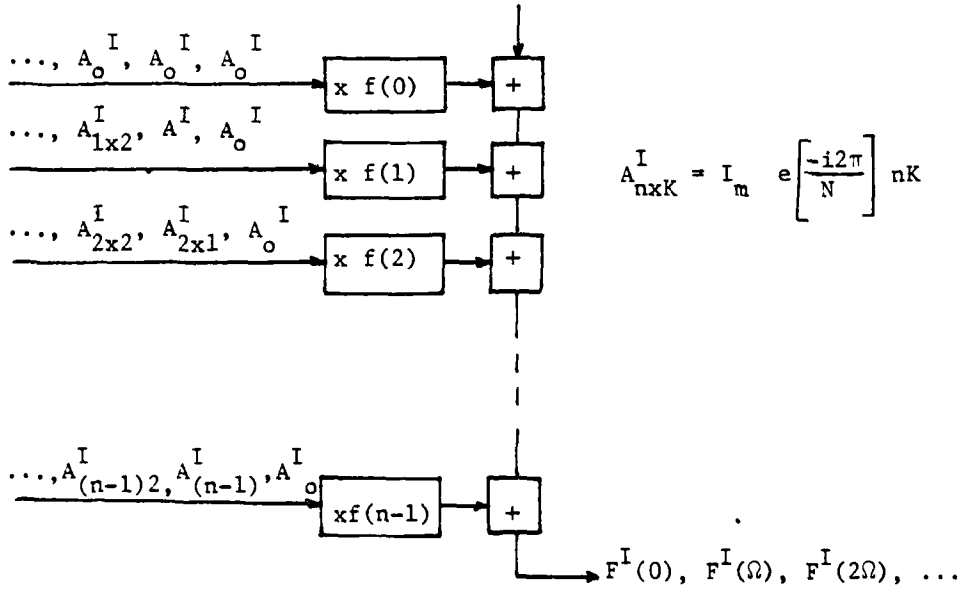
A X B (B + ic)

= AB + iAC



REAL

IMAGINARY



NOTE THAT VALUES FOR A_{NK} HAVE TO BE SCALE UP BEFORE COMPUTATION.

COMPUTATION TIME = N SET-RESET TIME OF MODULE

FIGURE 4.20 COMPUTATION OF DFT WITH REAL INPUT.

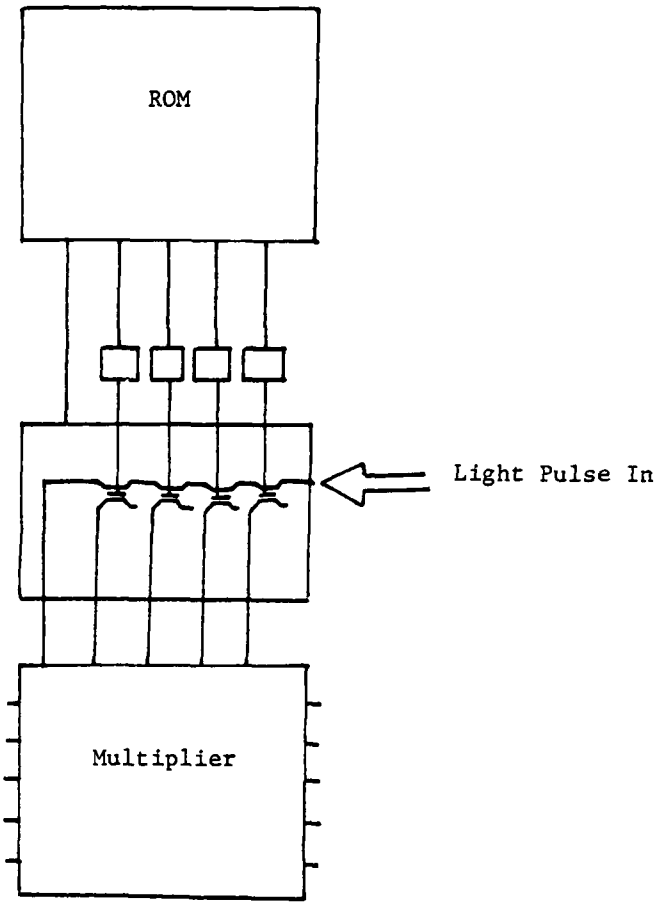
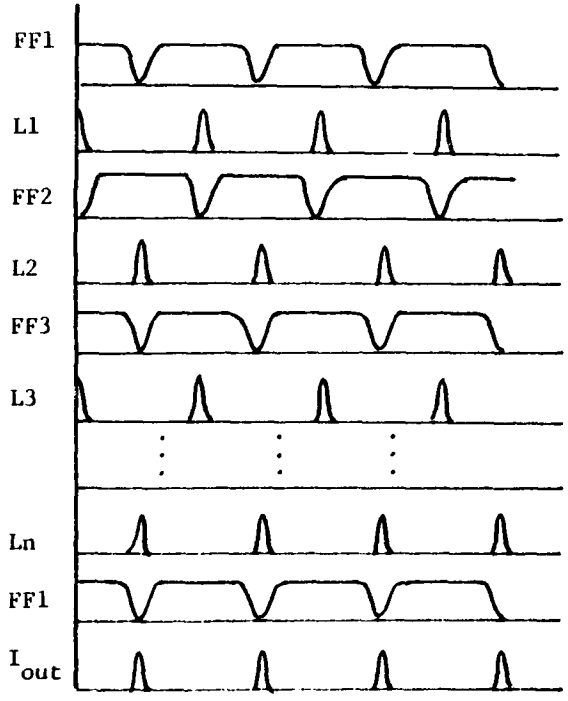


FIGURE 4.21 CONVERSION OF ELECTRONIC ROM OUTPUT INTO OPTICAL SPATIAL SIGNAL.



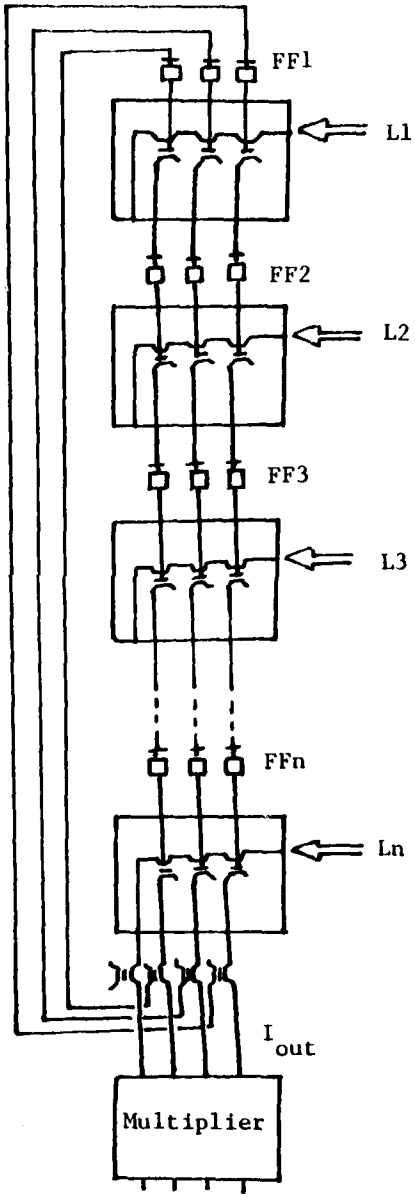


FIGURE 4.22 OPTICAL SHIFT REGISTER.

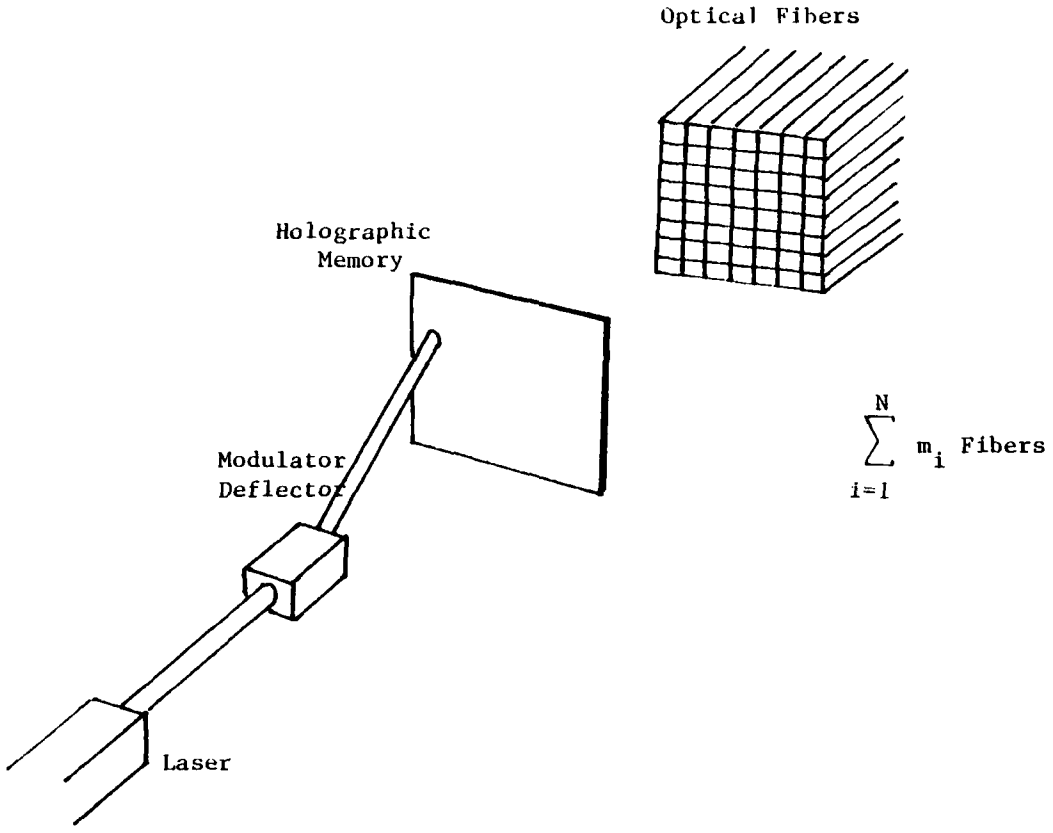
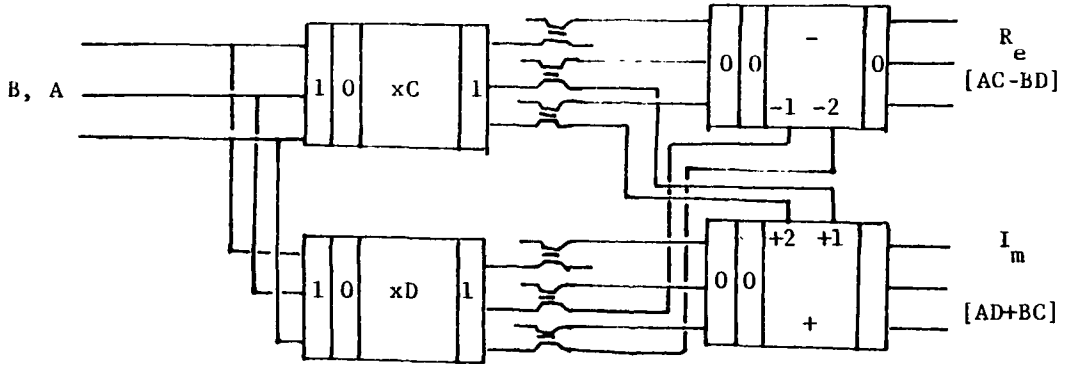


FIGURE 4.23 HOLOGRAPHIC READ-ONLY MEMORY.

involves 4 real multiplications, 1 addition and 1 subtraction. Instead of using separate modules for each operation, the real and imaginary parts can be time-multiplexed to reduce the number of required hardwares. The process is illustrated in Figure 4.24 for $(A + iB)\cdot(C + iD) = [AC - BD] + i[AD + BC]$. The multipliers are programmed for xC and xD operations. The real part A is entered into the multipliers first, setting the subsequent modules for -AD and +AC operations. The imaginary part B then propagates through both multipliers and the adder-subtractor modules. The light beam exiting from one of the module will give the real part of the output [AC - BD) while the other provides the imaginary part (AD + BC). The switches between the multipliers and adders will be alternatively switching the light exiting from the multipliers between the programming inputs and the operators inputs of the adder modules. The computation time of a complex multiplication would be twice that of a real multiplication.

Using this concept, the complex Fourier transformation can be performed as shown in Figure 4.25. An electronic ROM is used for the storage of the real and imaginary part of the coefficients. Though this may not be the optimum in terms of speed, it may be the more readily achievable approach. With the systems described above, the pipelining concept was not utilized. To perform a 1024 point DFT for example, the light pulse would have to propagate through 1024 adders. This would be very inefficient both in terms of optical loss and system speed. To pipeline the system, we can use the cascaded arrangements as shown in Figure 4.19. However, a substantial number of additional adders would be necessary. Alternatively, we can make use of the fact that the input f(nT) are entered into the computer sequentially and utilize the system shown in Figure 4.26. With this system, the computation begins as the input data are filling up the multipliers instead of waiting till all N samples are obtained. As the second input data is entered to the second multiplier, the



$$[A + iB] \times [C + iD] = [AC - BD] + i[AD + BC]$$

Figure 4.24. Computation of Complex Arithmetic

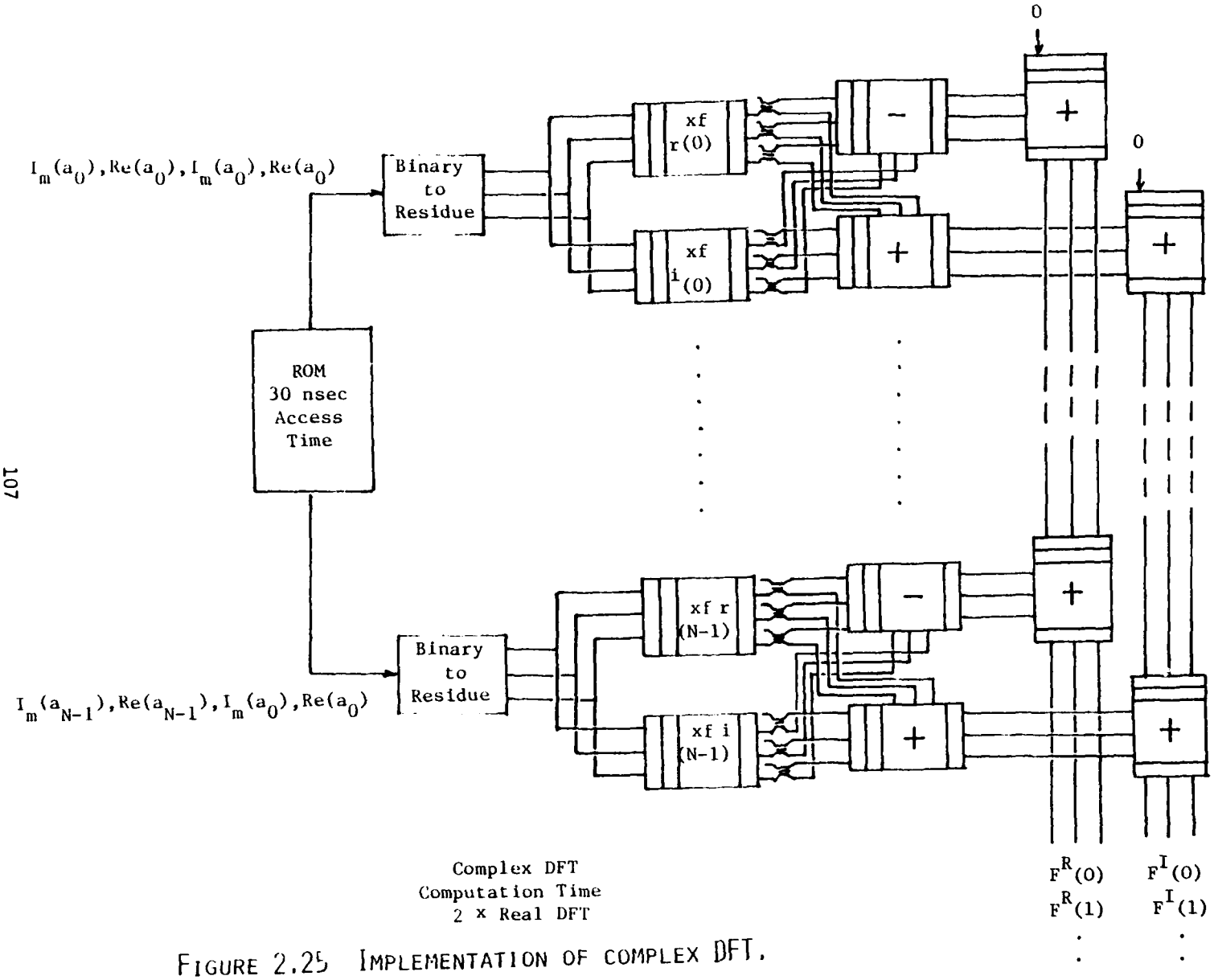
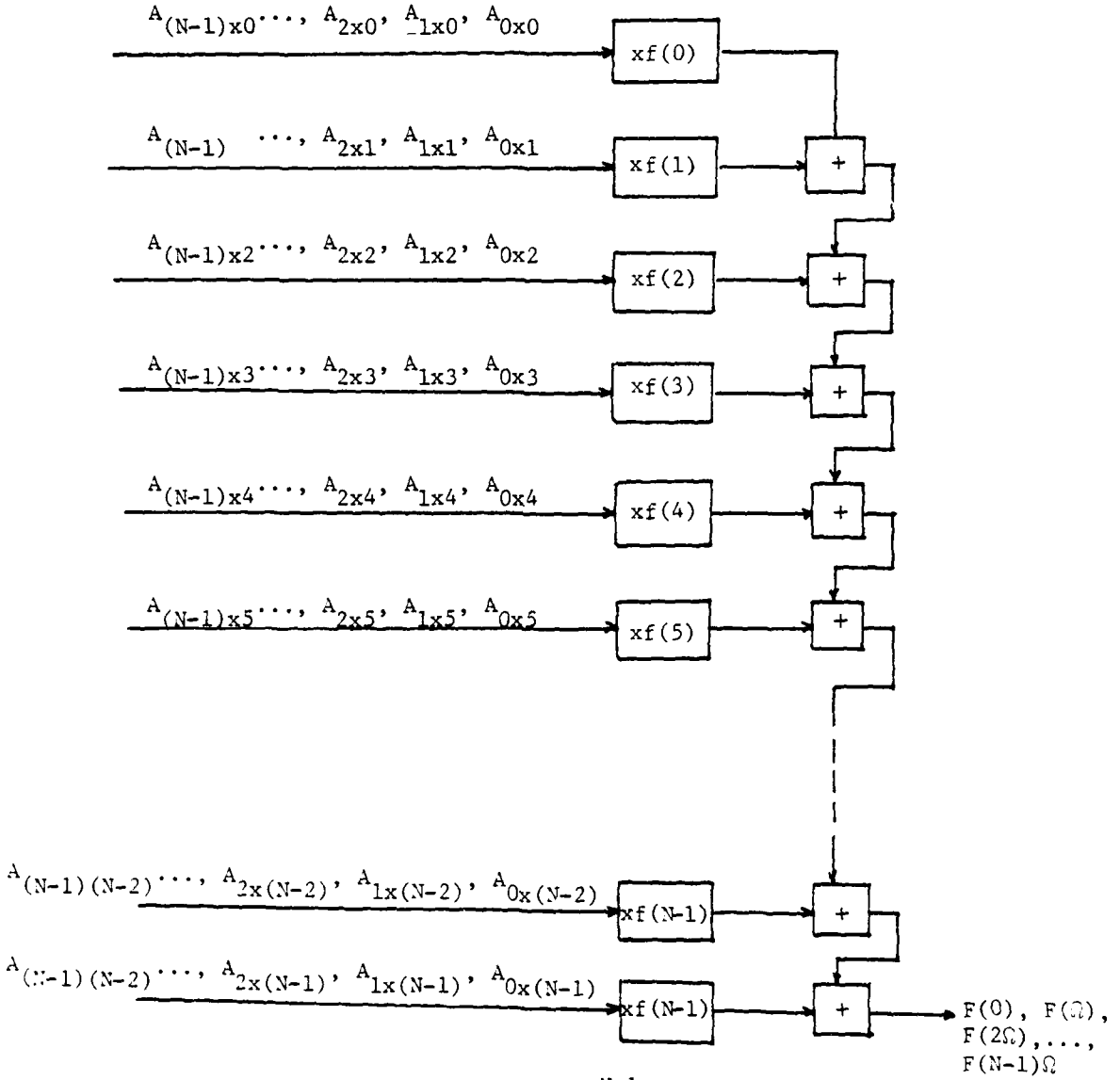


FIGURE 2.25 IMPLEMENTATION OF COMPLEX DFT.



$$F(K\Omega) = F\left(K\frac{2\pi}{NT}\right) = \sum_{n=0}^{N-1} f(nT)A_{kxn}$$

$$A_{Kxn} = e^{-1\frac{2\pi}{N}Kn}$$

FIGURE 4.26 PIPELINING OF DFT COMPUTATION.

first coefficient for the first input is propagated through the multiplier and the first adder is set. When the third input data is entered, the first coefficient of the second input is propagated through both the second multiplier and the first adder setting the second adder. The first adder is then reset. When the fourth input is entered, the second coefficient of the first input and the first coefficient of the third input would be sent through their respective multipliers to set the next adder. The time sequence of the system operation is shown in Figure 4.27. The data input is entering the computing system at twice the computation rate. One clock cycle after the last data f[(N-1)T] is entered, the values of the transform f(kΩ) would begin to flow out of the last adder. Since the light pulses pass through only two modules in each cycle, with the assumption of 3 nsec set-reset time and 40 psec propagation time for the computation modules, the cycle time would be 3.08 nsec. For a 1024 point DFT with real input functions, the transform can be completed in about 3.2 μsec after the last input data is entered.

4.6.2 FFT

The introduction of the Fast Fourier Transform technique has greatly reduced the computation time for digital Fourier transform. The same benefit can be realized for the optical residue computer.

There are two basic techniques, decimation in time and decimation in frequency. With decimation in time, the original sequence of N numbers is subdivided into 2^k sequences. For illustration, let us examine the case where k = 1. A sequence f(n) with n=0,...N-1, is divided into two sequences g(n) and h(n) where

$$g(n) = f(2n)$$

with

$$n = 0, \ldots \frac{N}{2} - 1$$

and

$$h(n) = f(2n + 1)$$

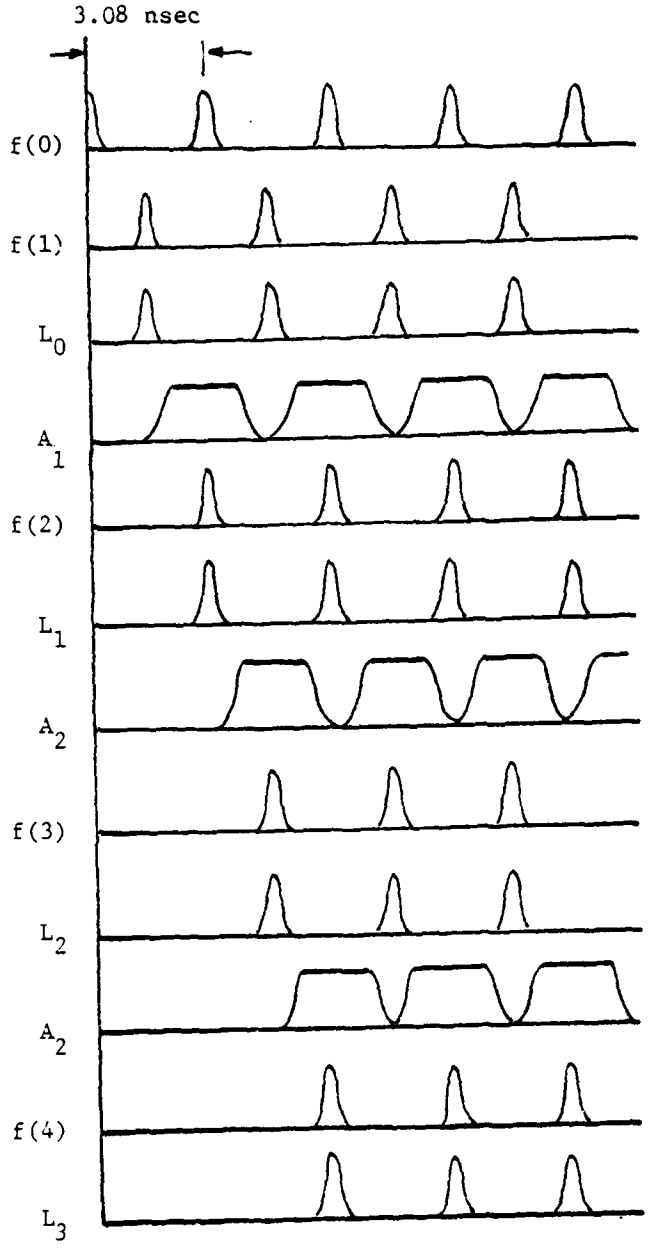


FIGURE 4.27 TIMING SEQUENCES OF PIPELINED DFT COMPUTATION.

The Fourier transform of f(n) can be written as

$$F(k) = \sum_{n=0}^{N/2-1} g(n)e^{-1\frac{2\pi}{N}2nK} + e^{-1\frac{2\pi}{N}}\sum_{n=0}^{\frac{N}{2}} h(n)e^{-1\frac{2\pi}{N}nK} \qquad (4.6)$$

The N point DFT of f(n) can therefore be expressed as the summation of two $\frac{N}{2}$ point DFT. These two $\frac{N}{2}$ point DFT however, can be performed simultaneously in parallel. The computation time would therefore be reduced to that of a $\frac{N}{2}$ point DFT. The FFT algorithm can be implemented with the optical residue computer concept for real input as shown in Figure 4.28. With this arrangement, the upper half will compute one cycle ahead of the lower half. The output of G(k) would set the adders while the multipliers are set for the coefficients $e^{-i\frac{2\pi}{N}K}$ and $e^{-i\frac{2\pi}{N}K+1}$. At the next cycle the output of H(k) would propagate through the two multipliers and adders to produce the values of f(k) and $f(\frac{N}{2}+k)$.

With this system, the pipelined arrangement shown in Figure 4.26 for the computation of DFT cannot be utilized if we want to fully maintain the speed advantage. The reason is that the computation can not begin with FFT until all N input data are entered. The cascaded version shown in Figure 4.19 can be used but it would require additional hardwares and add $\log_2 \frac{N}{2}$ cycles to the total computation time. With such a system, the transform can be completed in $\frac{N}{2} + 1 + \log_2 \frac{N}{2}$ cycles. For a 1024 point input, this would correspond to 1.6 μsec. The computation time is therefore cut almost by half.

An alternate technique is decimation in frequency. With this technique, the input data sequence is also subdivided into 2^k sequences. Once again, let us examine the case where k = 1. An input sequence f(n) can be divided into two sequences g(n) and h(n) where

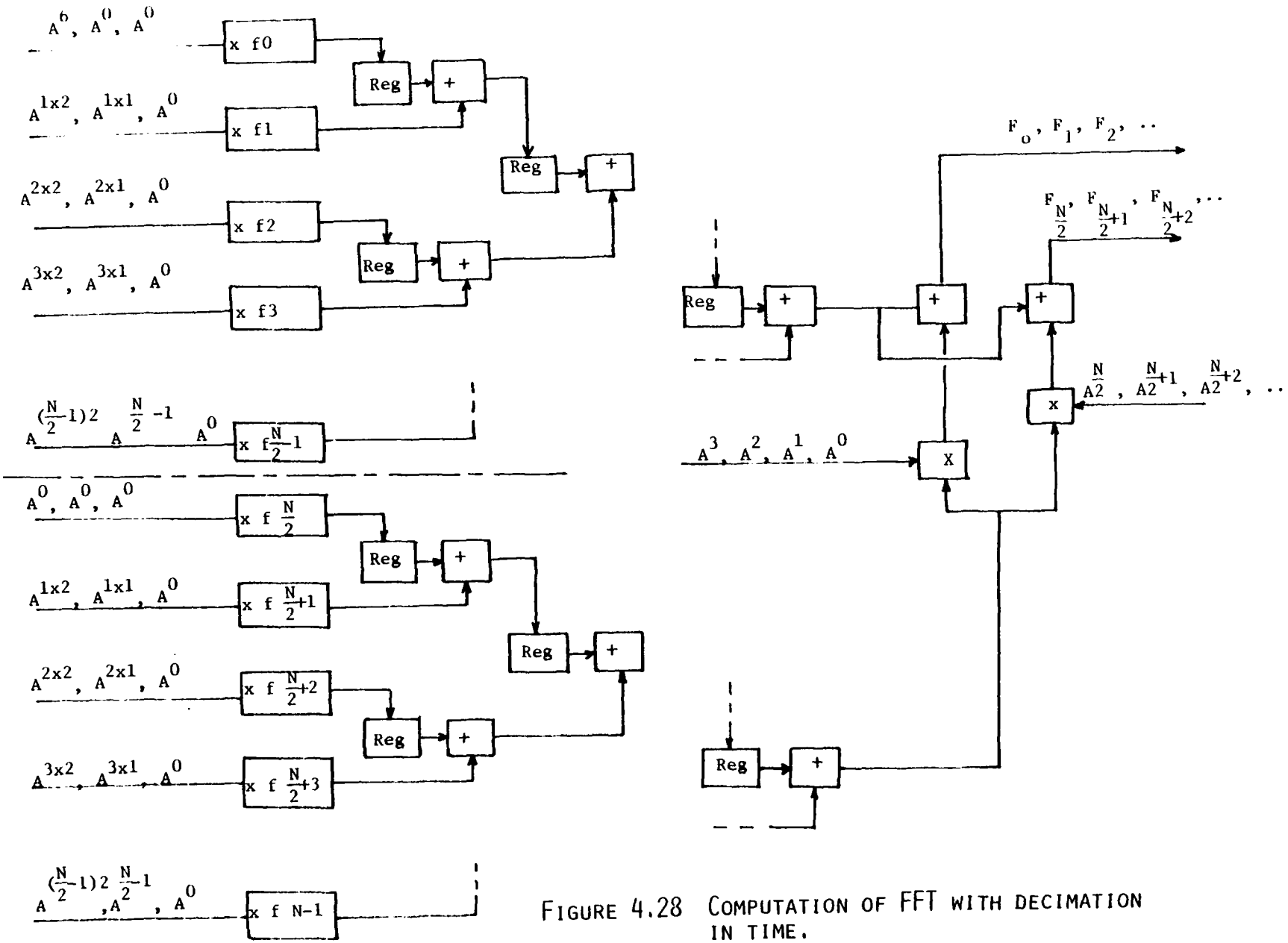


FIGURE 4.28 COMPUTATION OF FFT WITH DECIMATION IN TIME.

$$g(n) = f(n)$$

for $$n = 0, \ldots \frac{N}{2} - 1$$

$$h(n) = f(n + \frac{N}{2})$$

The Fourier transform of f(n) can then be written as

$$F(k) = \sum_{n=0}^{N/2-1} g(n)e^{-i\frac{2\pi}{N}nK} + h(n)e^{-i\frac{2\pi}{N}(n + \frac{N}{2})k}$$

$$= \sum_{n=0}^{\frac{N}{2}-1} \left(g(n) + e^{-i\pi k}\, h(n)\right)e^{-i\frac{2\pi}{N}nk} \qquad (4.7)$$

Separating the odd and even points of F(k) and letting 2m = k, we have

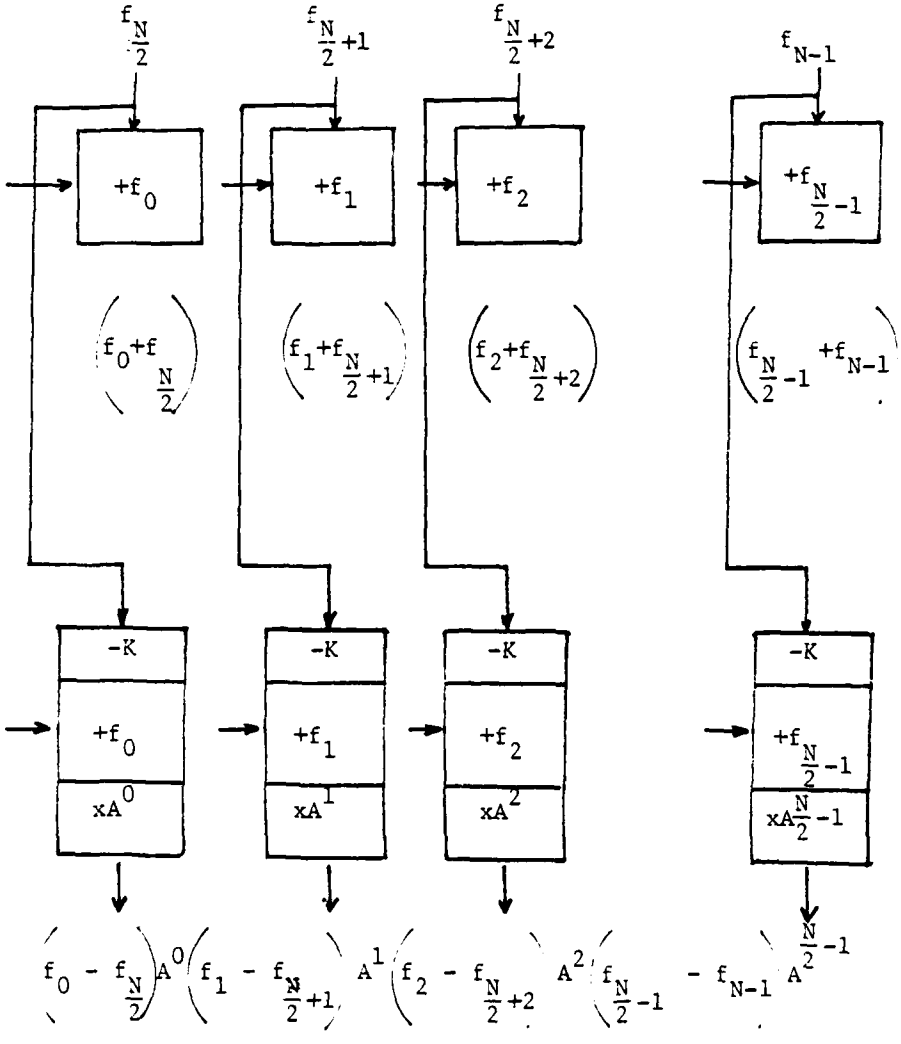
$$F(2m) = \sum_{n=0}^{N/2-1} (g(n) + h(n))\; e^{-i\frac{2\pi}{N}\,2nm} \qquad (4.8)$$

and

$$F(2m + 1) = \sum_{n=0}^{N/2-1} (g(n) - h(n))e^{-i\frac{2\pi}{N}n}\; e^{-i\frac{2\pi}{N}2nm} \qquad (4.9)$$

What we have is essentially two N/2 point DFT for the functions.

$$(g(n) + h(n)) \text{ and } (g(n) - h(n))e^{-\frac{2\pi}{N}\ell}$$



$$A^n = e^{-i\frac{2\pi}{N}n}$$ (scaled up and rounded off)

FIGURE 4.29 PRIOR COMPUTATION FOR FFT WITH DECIMATION IN FREQUENCY.

To implement this algorithm, we would require N additional computation modules to compute the values of

$$(g(n) + h(n)) \text{ and } (g(n) - h(n))e^{-i\frac{2\pi}{N}\ell}$$

as shown in Figure 4.29. The output are then entered into the multipliers similar to the arrangement for DFT in Figure 4.30. The computation time is slightly faster than the system using decimation in time, requiring $\frac{N}{2} + 1$ cycles. For a 1024 point FFT, the computation time would be 1.58 μsec. The number of computation modules required to implement the decimation in time system is

$$2N + 4 + \frac{N}{2} + \frac{N}{4} + \frac{N}{8} + \ldots 1 = 3075$$

While for the decimation in frequency system, the number of modules is 3N = 3072. Thus we see that the efficiencies of both techniques are about equal. With this first order of decimation, the computation time is decreased by about 100% with 33% more hardwares. Additional increase in computation speed can be realized with further decimation The computation time is approximately equal to $\frac{2^k}{2N}$ cycles where 2^k is the number of input data point and N is the number of times the input sequence is subdivided. Since a cycle is equal to one set-reset time of a module plus the propagation time through two modules the cycle time is about 3.08 nsec . To maintain this throughput rate, an optical ROM using the cyclically shifting data registers as shown in Figure 4.22 must be used.

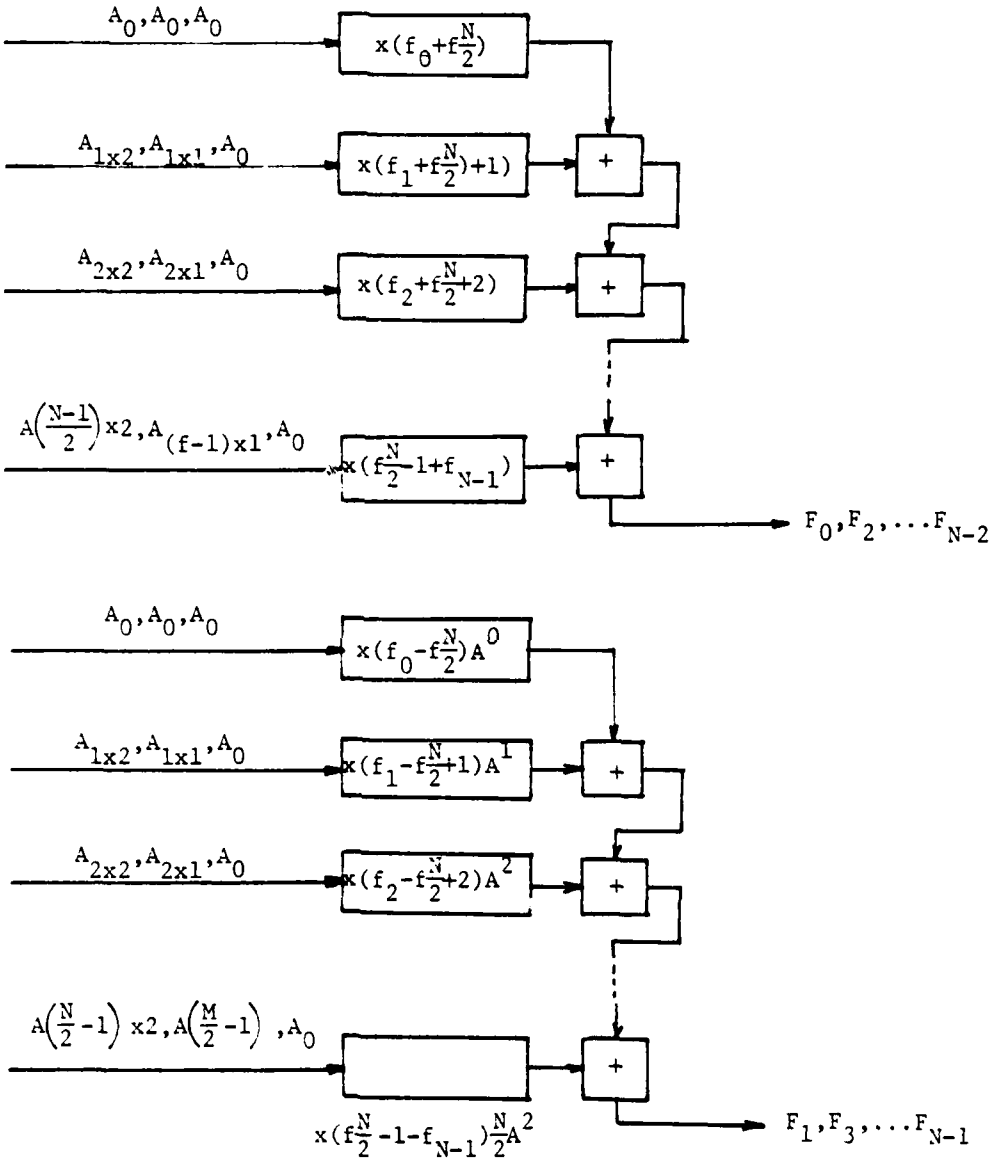


FIGURE 4.30 PIPELINED COMPUTATION OF FFT WITH DECIMATION IN FREQUENCY.

4.6.3 TWO-DIMENSIONAL DFT

The same concept can be extended to two-dimensional Fourier transform. The DFT of a two-dimensional input function f(nx,my) with n = 0, N - 1 and m = 0, M-1 can be written as

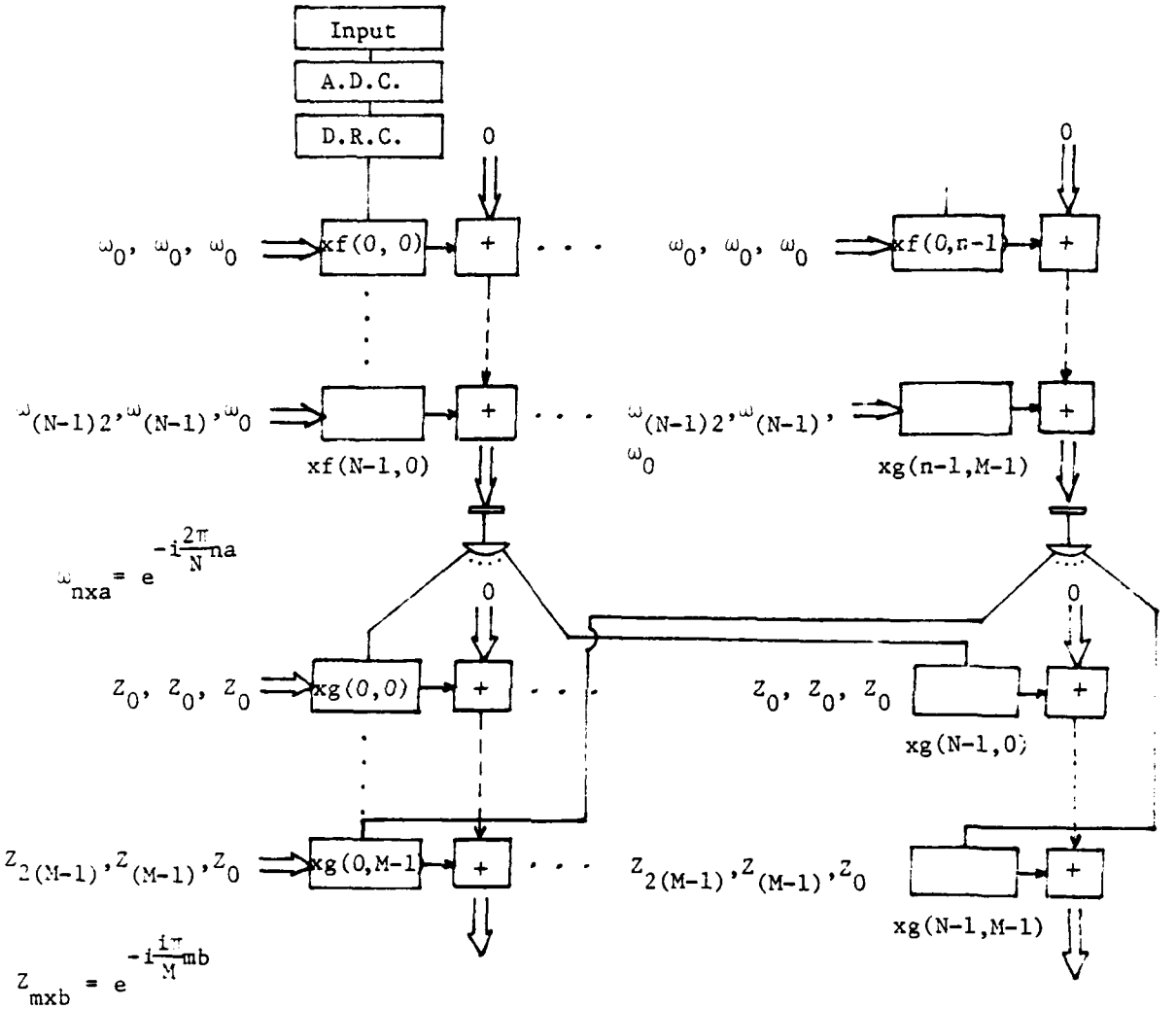
$$F(ap,\ bq) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(nx,my) e^{-inapx}\ e^{-imbqy}$$

$$= \sum_{m=0}^{M-1} q(ap,my) e^{-imbqy} \qquad (4.10)$$

where

$$p = \frac{2\pi}{Nx} \text{ and } q = \frac{2\pi}{My}$$

To optimize speed, the highly parallel structure shown in Figure 4.31 can be used. The output of the first transform operation g(ap,my) would program a second set of multipliers. The system can be pipelined in such a way that as the first column of multipliers are being programmed by the output g(0,m) the computation begins for the second dimension for the values of F(0,m). With such a high degree of parallism, the computation time for a N x N input would only be 2N cycle times. The number of required computatior modules would be $2(2N)^2$. Thus if N is large, such an arrangement would not be practical. Alternatively, the system structure shown in Figure 4.32 can be used to reduce the number of hardwares at the expense of computing speed. With this arrangement, the multipliers have to be reprogrammed after the computation for each column of input data. The output are stored in a buffer storage which is then loaded into the second set of

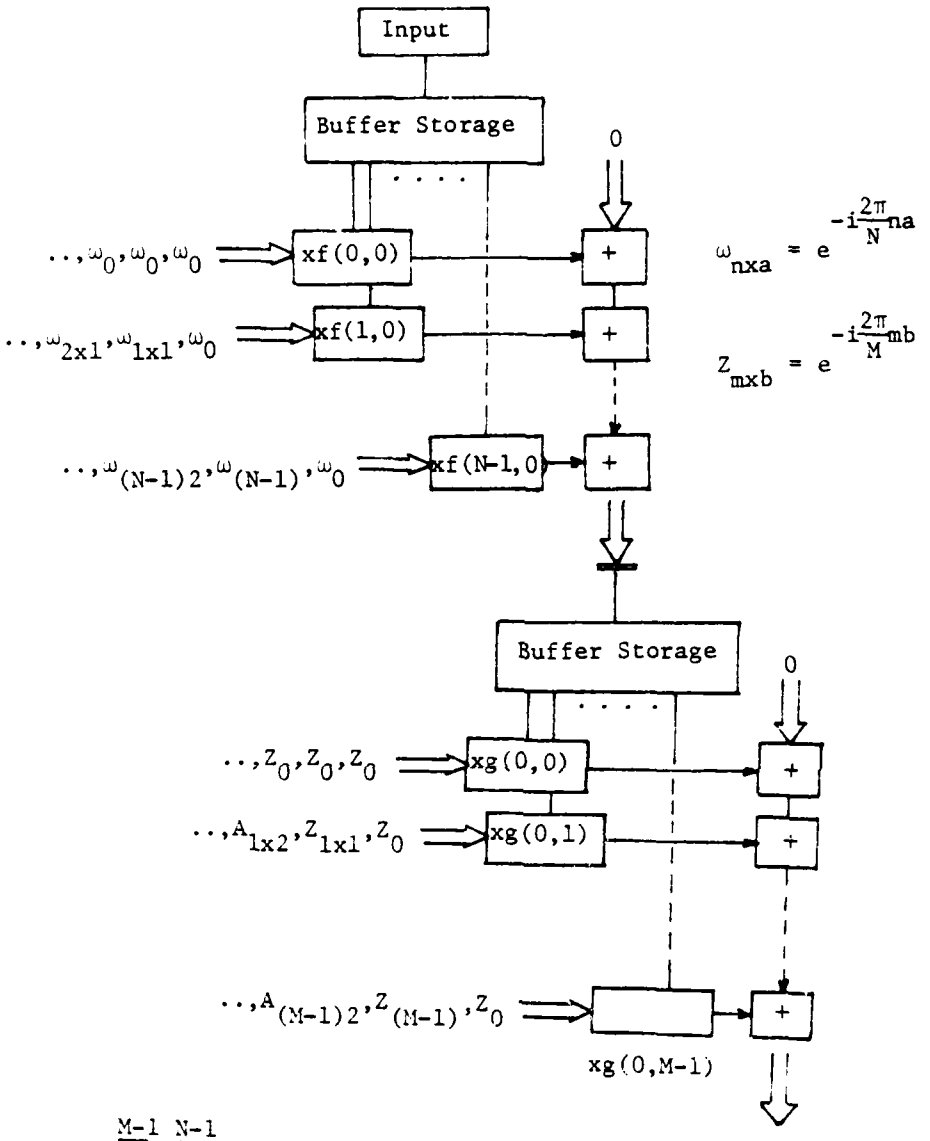


F(0, 0) . . . . . F(N - 1, 0)
F(0, 1) . . . . . F(N = 1, 1)
⋮ ⋮
F(0, M - 1) . . . . . F(N - 1, M - 1)

$$F(ap, bq) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(nx, my)\, e^{-inapx}\, e^{-mbq\, y} \qquad p = \frac{2\pi}{Nx}$$

$$= \sum_{m=0}^{M-1} g(ap, my)\, e^{imbqy} \qquad q = \frac{2\pi}{My}$$

FIGURE 4.31 PARALLEL COMPUTATION OF 2-DIMENSIONAL DFT.



$$F(ap, bq) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(nx, my) \cdot e^{-inapx} e^{-imbqy}$$

$$= \sum_{m=0}^{M-1} g(ap, my)\, e^{-imbqy}$$

FIGURE 4.32 SERIAL COMPUTATION OF 2-DIMENSIONAL DFT.

multipliers after the computation for the first dimension has been completed. The transform operation for a N x N input signal would require at least $2N^2$ cycles. However, it would require only 4N computation modules and 2N optical data registers for the buffer storage. We may also note that the discussions above would also apply to the extension of the convolution and correlation into 2-dimensional operations.

4.7 ENCODING

Before computation can be performed with the modules, the input must be encoded into its equivalent residue number in the appropriate spatial representation. The simplest approach may be to convert the analog input into an intermediate binary form with the use of a conventional A to D converter or the integrated optics implementation scheme introduced by Taylor[24]. The binary input can then be converted into residue numbers in the spatial form with the arrangement shown in Figure 4.33. We note that for modulus 5,

$$2^0 = 1,\ 2^1 = 2,\ 2^2 = 4,\ 2^3 = 3$$

$$2^4 = 1,\ 2^5 = 2,\ 2^6 = 4,\ 2^7 = 3$$

For example

$$58 = 1\ \ 1\ \ 1\ \ 0\ \ 1\ \ 0$$

$$|58|_5 = |1 \times (2^5) + 1 \times (2^4) + 1 \times (2^3)$$

$$+ 0 \times (2^2) + 1 \times (2^1) = 0 \times (2^0)|_5$$

$$= |1 \times 2 + 1 \times 1 + 1 \times 3 + 0 \times 4$$

$$+ 1 \times 2 + 0 \times 1|_5$$

$$= |2 + 1 + 3 + 2|_5 = 3$$

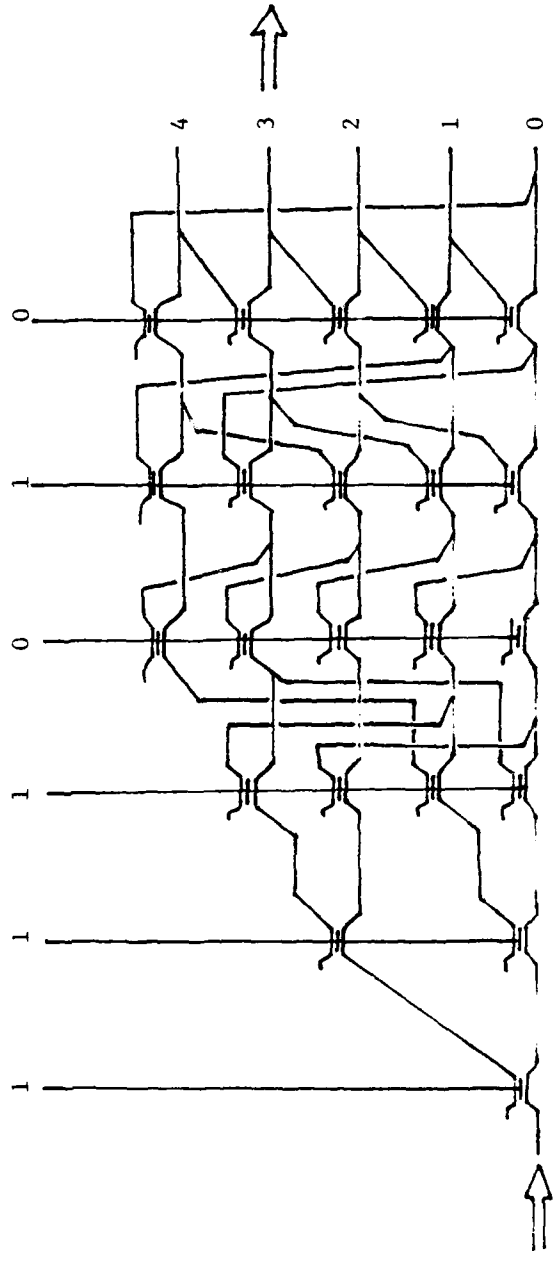


FIGURE 4.33 BINARY TO RESIDUE CONVERTER.

Thus, we see that encoding can be performed in 1 set time of the module. We may note that the binary to residue conversion can also be implemented using the programmable modules as shown in Figure 4.34.

The same concept can be modified for decimal to residue conversion by noting that an integer

$$123 = 1 \times 100 + 2 \times 10 + 3$$

The conversion procedure requires three steps. The decimal digits are individual to converted into into the residu equivalent. It involves a 10 to m_i mapping which can be achieved with the use of mixed maps. The residue digits are then multiplied by $x|10^n|_{m_i}$ which can also be performed with fixed maps. Then the product for each residue digits are summed with modular additions to obtain the residue equivalent. The algorithm is illustrated in Figure 4.35 and the implementation is shown in Figure 4.36.

4.8 DECODING

Decoding a residue number is a more complicated operation than encoding. The most popular approach is to convert the residue number into the mixed radix system[2]. The reason is that the conversion procedure can be performed with the same type of hardware used for the basic residue arithmetic computations. The algorithm is shown schematically in Figure 4.37 for moduli 2, 3, 5, 7, 11. $|1/K|_{m_i}$ represents the multiplicative inverse where $|K \times |1/K|_{m_i}|_{m_i} = 1$. The drawback is that the procedure requires N-1 sequential steps where N is the number of moduli used. Since encoding and computation can be performed at a throughput rate of 1/one set-reset time of a module, this sequential decoding procedure would seemingly be a bottleneck for the entire process. Fortunately, the conversion procedure can be pipelined. To pipeline the operation, it is

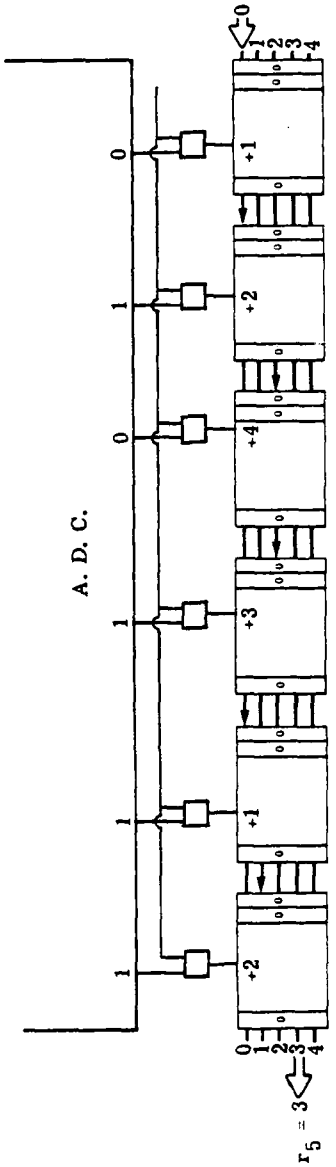


FIGURE 4.34 BINARY TO RESIDUE CONVERSION USING PROGRAMMABLE COMPUTATION MODULE.

$$123 = 1 \times (100) + 2(10) + 3$$

For Mod 7

MOD 2, 3, 5, 7

123 = [1, 0, 3, 4]

x =	1	2	3
r =	1	2	3
$x\lvert 10^n \rvert_7$ =	x2	3	1

$2 + 6 + 3 = 11 \Rightarrow$ [4]

10^n	10^2	10^1	10^0
$\lvert 10^n \rvert_7$	2	3	1

Mod 5

x =	1	2	3
r =	1	2	3
$x\lvert 10^n \rvert_5$ =	x0	0	1

$0 + 0 + 3 = 3 \Rightarrow$ [3]

10^n	10^2	10^1	10^0
$\lvert 10^n \rvert_5$	0	0	1

Mod 3

x =	1	2	3
r =	1	2	0
$x\lvert 10^n \rvert_3$ =	x1	1	1

$1 + 2 + 0 = 3 \Rightarrow$ [0]

10^n	10^2	10^1	10^0
$\lvert 10^n \rvert_3$	1	1	1

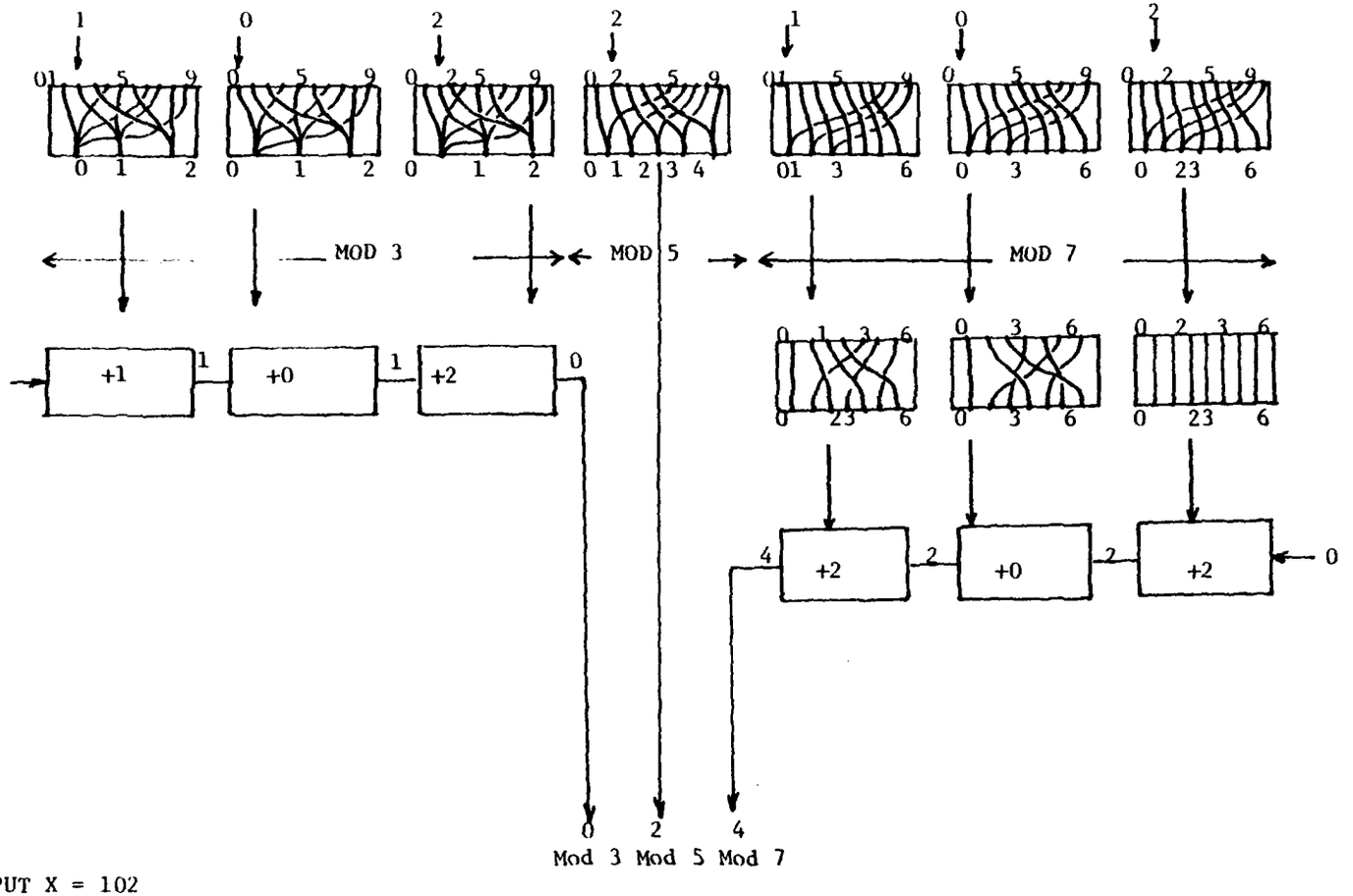
Mod 2

x =	1	2	3
r =	1	0	1
$x\lvert 10^n \rvert_2$ =	x0	0	1

$0 + 0 + 1 = 1 \Rightarrow$ [1]

10^n	10^2	10^1	10^0
$\lvert 10^n \rvert_2$	0	0	1

Figure 4.35. Decimal to Residue Conversion



INPUT X = 102

MOD 3, 5, 7 = (0, 2, 4)

CONVERSION TIME

= 1 SWITCHING TIME OF ADDER MODULE

FIGURE 4.36. DECIMAL TO RESIDUE CONVERSION.

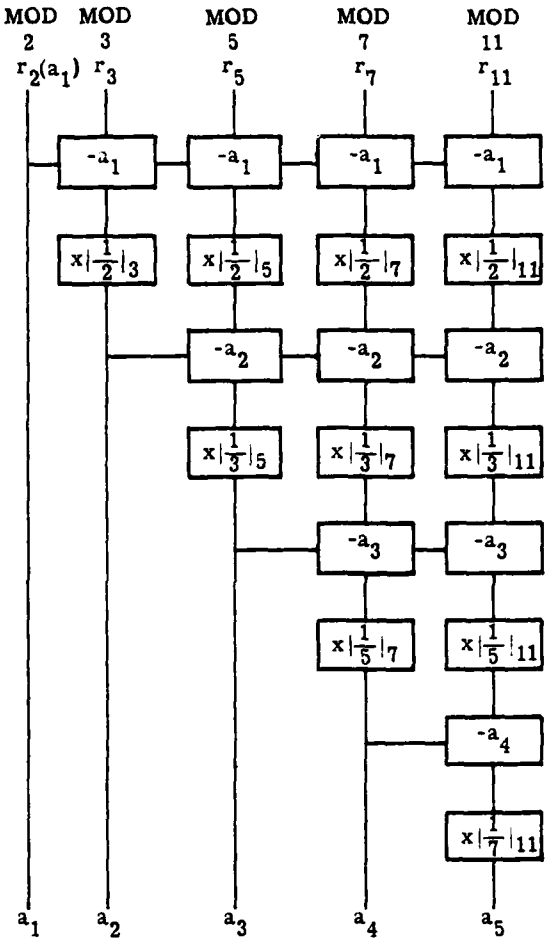


FIGURE 4.37 ALGORITHM FOR RESIDUE TO MIXED RADIX CONVERSION.

necessary to synchronously delay the coefficients obtained earlier in the procedure such that all the coefficients would advance through the decoding procedure at the same rate. This necessary delay can be accomplished by use of the simple data register module shown in Figure 4.38. We also note that the multiplying factors $|1/m_i|_{m_j}$ are fixed and they can be implemented by fixed maps. The design of a pipelined residue-to-mixed radix converter is illustrated schematically in Figure 4.39. The input residue numbers are first stored in the data register modules (represented by boxes with bold lines). At the same time, the computation modules are set by r_1 for the $-a_1$ operation. Light pulses are then injected into the data registers to recall the residue numbers. The light pulses propagate through the computation modules performing $-a_1$ operation, and then through the fixed maps for $x|1/2|_{m_i}$. The output is stored in the next set of data register modules and the next computation modules are set for the $-a_2$ operation. Simultaneously, the second entry of the residue numbers are entered into the first set of data registers, ready for the first step of computation. The timing sequence of the input, the data recall light pulses and the output are shown in Figure 4.40. We see that no part of the converter sits idle at any time and the conversion is performed at a constant throughput rate of 1/one set-reset time of a computation module. The pipelining concept can be applied to any sequential computation procedure. The encoding computation and decoding can therefore be performed at the same throughput rate. Assuming once again that the set-reset time of a computation module is 3 nsec and the propagation time is 40 psec, a numerical optical computer with a system throughput rate of 320 MHz would be possible.

Residue to mixed radix conversion is a very important procedure in residue arithmetic. Besides decoding the output, the conversion is used for other important operations[2] such as sign detection, magnitude comparison, and overflow detection. Pipelining the procedure is therefore an important concept in an optical numerical

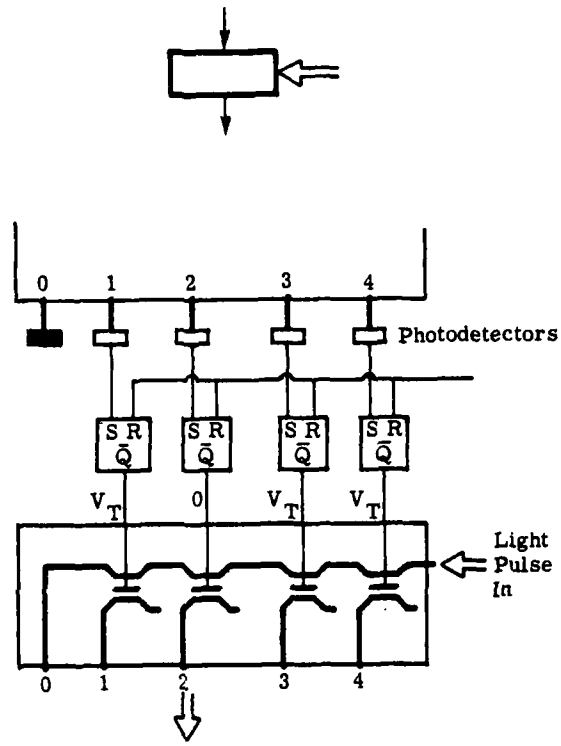


FIGURE 4.38 OPTICAL DATA REGISTER MODULE.

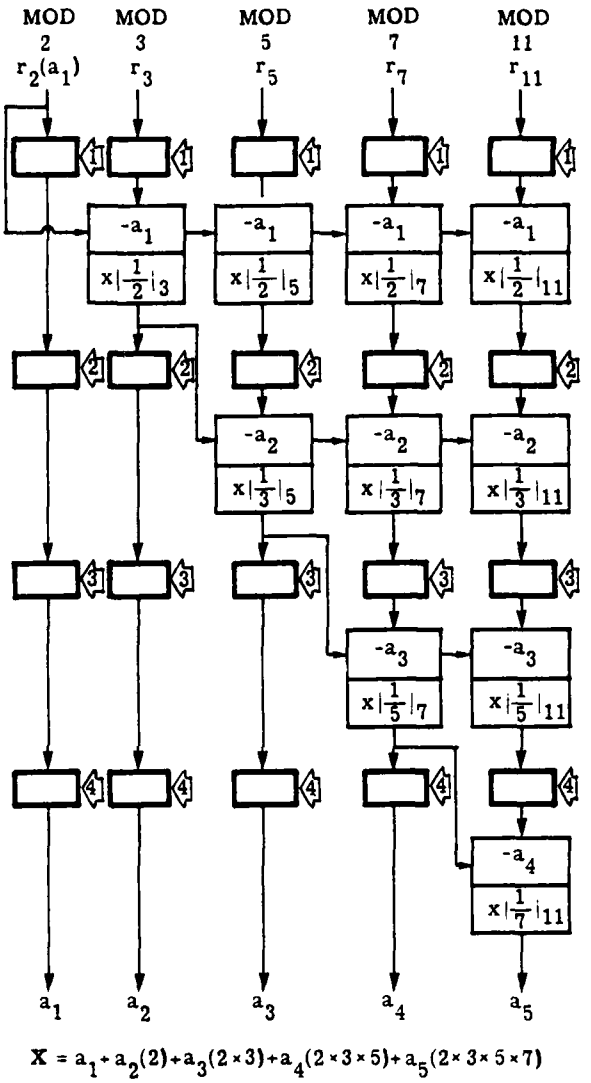


FIGURE 4.39 PIPELINED RESIDUE TO MIXED RADIX CONVERSION.

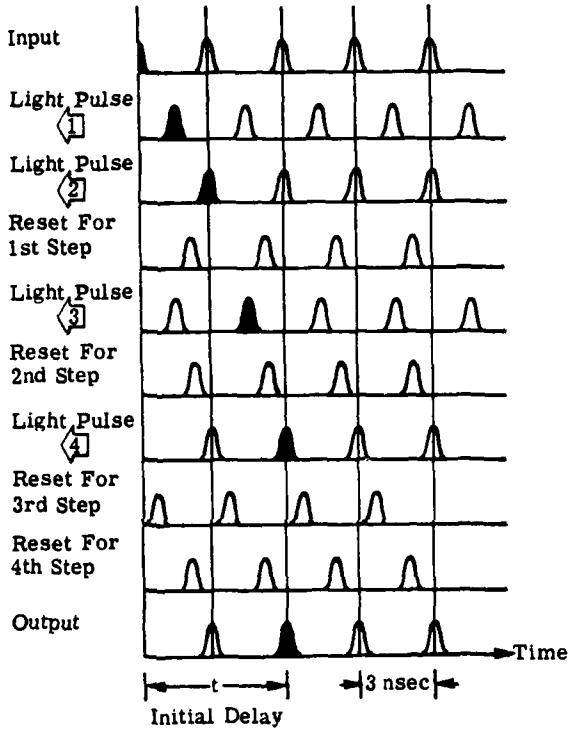


FIGURE 4.40 TIMING SEQUENCE FOR PIPELINED RESIDUE TO MIXED RADIX CONVERSION.

computer using residue arithmetic. The original residue number may be stored in a cascade of data registers while it is being converted into the mixed radix form for condition check. The residue number is moved down at the same rate as the conversion process and the computation is continued at the same rate as the conversion process and the computation is continued after the checking is completed. Alternatively, after the residue numbers are converted into their mixed radix equivalent for sign detection or overflow detection, they can be converted back into the residue form for further computation. The inverse conversion (mixed radix to residue) can be achieved very easily, and once again in one set time of the computation module.

Let us take the case where the moduli are 2, 3, 5, 7. The residue representation can be written as $x = (r_1, r_3, r_5, r_7)$ and the mixed radix representation as $x = [a_1, a_2, a_3, a_4] = a_1(1) + a_2(1 \times 2) + a_3(1 \times 2 \times 3) + a_4(1 \times 2 \times 3 \times 5)$. For example, to calculate the residue for modulo 3,

$$r_3 = |a_1|1|_3 + a_2|2|_3 + a_3|6|_3 + a_4|30|_3|_3.$$

Since $|30|_3 = |6|_3 = 0$, $r_3 = |a_1|1|_3 + a_2|2|_3|_3$. The implementation is illustrated in Figure 4.41.

We like to point out that the extension of base operation required for the scaling operation is essentially a modified residue to mixed radix conversion algorithm. The same pipelining technique can also be used for the extension of base and the throughput rate can therefore be maintained with the scaling operation.

In order that the optical computer can communicate with conventional electronic computers, the output of the optical residue computer has to be converted into binary form. The binary representation of X is $(b_0, b_1, b_2, \ldots)$ where

$$X = b_0 + b_1 \cdot 2 + b_2 \cdot 2^2 + b_3 \cdot 2^3 + \ldots$$

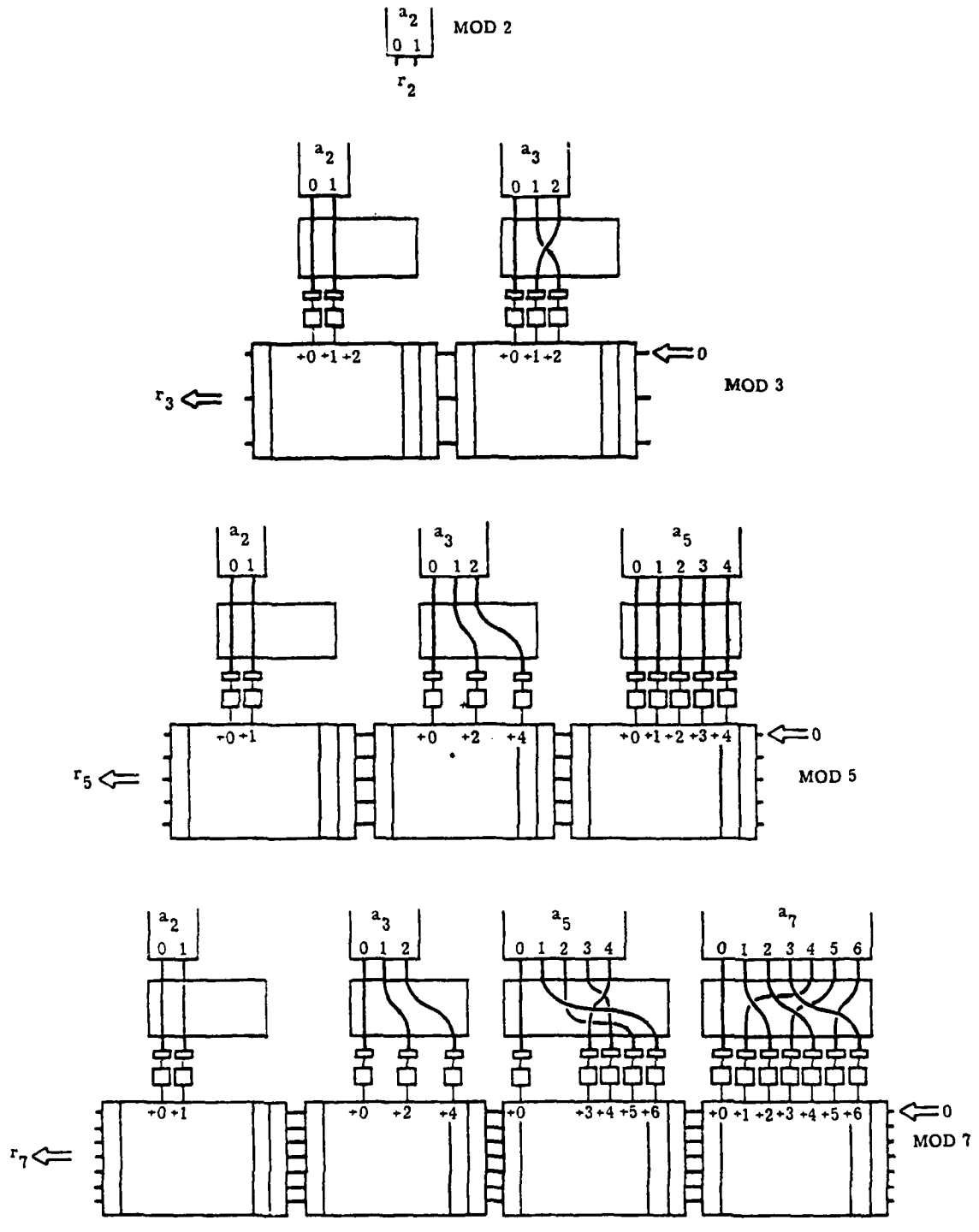


FIGURE 4.41 MIXED RADIX TO RESIDUE CONVERSION.

If one of the moduli of a residue representation is 2, then

$$b_0 = r_2$$

Subtracting b_0 from X, the result is divisible by 2. Thus, we can readily compute $(X - b_0)/2$ using fixed maps in the residue number system for all moduli except 2, for which the extension of base technique (which can be pipelined) must be used. Then b_1 is given by the new value of r_2. In a similar way all the binary digits b_i can be computed using the residue number system. However, the extension of base technique must be used for each binary digit (as opposed to just once for residue to mixed radix conversion). The entire procedure can be pipelined to keep the same dat a rate, but the initial time delay is roughly 3 nsec times the number of moduli times the number of bits in the number being decoded.

The above procedure for residue to binary conversion can be speeded up by working with a power of two for a modulus. For example, consider the case of modulus 8. Then

$$b_0 + b_1 \cdot 2 + b_2 \cdot 2^2 = r_8$$

Given r_8, the values of b_0, b_1, and b_2 can be easily determined by the simple look-up table structure shown in Figure 4.42 (a fixed map position to binary mapping). The number $(X - r_8)$ is divisible by 8 and so $(X - r_8)/8$ can be easily computed by residue arithmetic except for modulus 8, which must be done by the extension of base technique. Then $b_3 + b_u \cdot 2 + b_5 2^2$ is given by the new value of r_8, and b_3, b_4, and b_5 can be computed using the converter shown in Figure 4.42. This method requires only one-third the number of base extensions as compared with using modulo 2.

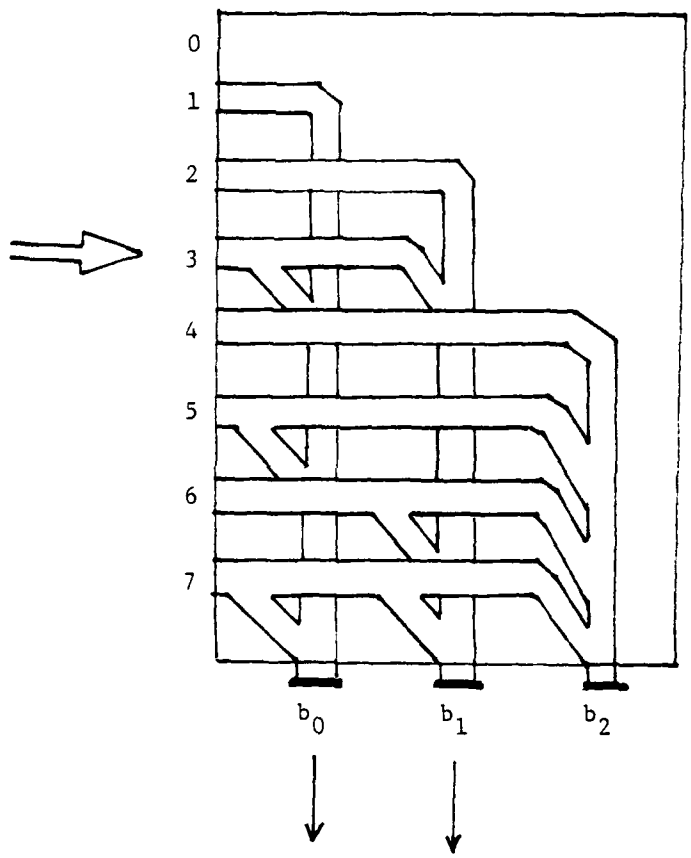


FIGURE 4.42 RESIDUE TO BINARY CONVERSION.

4.9 OPTIMIZATION OF COMPUTATION MODULE

In the first part of this section, we developed a programmable multipurpose computation module. The goal of that design is versatility However, we have also shown that the bit error rates and computation speed are greatly affected by the size of the computation module. More specifically, the optical loss and propagation delay through a module is proportional to the number of optical switches that the light pulses have to pass through. In this section, we shall take the opposite approach: instead of driving for maximum versatility, the computational modules are optimized in terms of the number of required optical switches. Such modules would be important to applications where speed and size are of prime importance while programmability is not a concern. These would include many special purpose signal processing applications.

We shall first examine the basic mod 7 adder shown in Figure The first order of switch reduction can be achieved by noting that the $+(S_1-S_2)$ operation is performed when $+S_1$ and $+S_2$ controls of the adder are activated. For example, for the modulo 7 adder, the +5 operation is performed when +6 and +1 controls flip flops are activated. The rows of switches for the operation of +5, +3, and +2 can therefore eliminate. With such a scheme, the number of optical switches required to implement the adder is reduced from $m_i(m_i-1)$ to $m_i(2\sqrt{m_i} - 2)$. More important, the number switches the light pulses have to be propagated through is reduced from (m_i-1) to $(2\sqrt{m_i} - 2)$. For example, with modulus 31, the probation delay per module is cut by 2/3 and the optical loss is reduced 26 times.

More generally, if a number of stages in the configuration s_1, s_2, ... are turned on, then the resultant operation can be shown to be

$$+a = +(s_1 - s_2 + s_3 - s_4 + \ldots)$$

where

$$s_1 > s_2 > s_3 \ldots$$

That operation +1 to +(m - 1) can then be accomplished by the N stages

$$s_{N-i+1} = 2^i - 1, \; i = 1, 2, \ldots, N$$

where

$$N = [\log_2 m]$$

This number of stages is far less than the number required by the conventional adder, m - 1. For example, operations +0 to +31 can be accomplished with the N = 5 stages +1, +3, +7, +15, and +31, as seen in Table 1. (Table 1 also shows the operations +0 to +15 using the four stages +1, +3, +7, +15, and so on.) The adders for modulus 5 using this scheme are shown in Figure 4.43. The routing of the waveguides within the adder is determined as follows. Start with a conventional module and eliminate all the switches except those for $+s_1$, $+s_2$, $+s_3$, etc., and simply straighten out the resulting paths where possible. The result is that a right-hand path exiting stage s_k will connect to the right-hand path entering stage s_{k+1} at a residue position (value $+|s_k - s_{k+1}|_m$ with respect to that at s_k. For modulus 5, $|s_1|_5 = |7|_5 = +2$.

A disadvantage of this scheme is that the +5 operation requires the input control light to be split into 3 parts, the +10 operation requires 4 parts, and the +21 operation requires 5 parts, etc. Thus

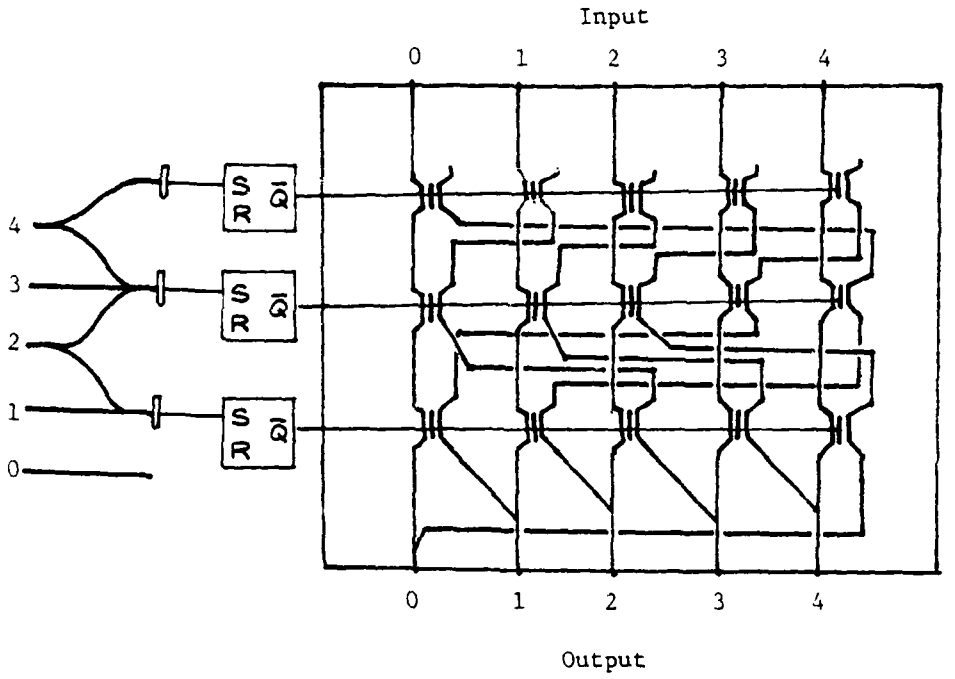


FIGURE 4.43 MODULO 5 ADDER WITH SWITCH REDUCTION.

TABLE 4.1 SETTING OF MULTIPLE STAGES REQUIRING THE MINIMUM TOTAL NUMBER OF STAGES

OPERATION A	STAGES SET (Contribution from each stage is + or -) 31	15	7	3	1
+31	+				
30	+				-
29	+			-	+
28	+			-	
27	+		-	+	
26	+		-	+	-
25	+		-		+
24	+		-		
23	+	-	+		
22	+	-	+		-
21	+	-	+	-	+
20	+	-	+	-	
19	+	-		+	
18	+	-		+	-
17	+	-			+
16	+	-			
15		+			
14		+			-
13		+		-	+
12		+		-	
11		+	-	+	
10		+	-	+	-
9		+	-		+
8		+	-		
7			+		
6			+		-
5			+	-	+
4			+	-	
3				+	
2				+	-
1					+
+0					

the reduction of the required number of stages (and switches) and the reduction of light losses due to switches is accomplished at the expense of light loss at the input controls. This trade-off will be favorable if a light pulse must pass through a number of modules before being detected as with the sum of products operation.

Similar switch reduction can be realized for the multiplier module. As shown in Figure 4.7 , a modulo m_i multiplier can be constructed from a modulo m_{i-1} adder. Thus, the minimum number of required switches for a modulo m_i multiplier would be

$$m_i \log_2(m_{i-1}) + (m_{i-1})$$

5
DESIGN CONCEPT ANALYSIS

5.1 INTRODUCTION

In the last section, we have presented a specific design for an optical residue computer using the mapping approach. We shall now examine the possible performance levels of such a system using demonstrated hardware performances. We shall emphasize the word 'demonstrated' since some of the hardware technologies are not yet in the production stage. The estimates would also apply only to the particular system presented and it is not to be taken as the inherent system performance of an optical residue computer. The implementation approaches reflect the present stage of component technology. The system design concepts will evolve together with the development of component technology.

With the performance estimates, the optical residue computer will then be compared with electronic units having similar capabilities. The purpose of this comparison is solely to provide some perspectives for the performance levels of the optical computer. Performance is a function of complexity but the complexities of an optical and an electronic system cannot be compared directly or fairly due to the vast differences in the stages of development and design concepts. Thus whatever comparisons that are made between the optical and electronic units should be taken in relative terms.

5.2 DEMONSTRATED HARDWARE PERFORMANCES AND PERFORMANCE LEVEL ESTIMATES FOR OPTICAL RESIDUE COMPUTER

The setting and resetting of a computation module require four sequential steps: detection, setting the flip flops and couplers, propagating a light pulse through the module and the resetting of the flip flops and couplers. The architecture of the optical computer is designed to minimize the number of such setting and resetting operations by using parallel structure. When a sequence of set-reset operations are required, pipelining is utilized to maintain the throughput rate of approximately 1/one-set-reset time of the computation module.

To achieve a 300 MHz throughput, a set-reset cycle time of about 3 nsec is required for the computation module. In Figure 5.1, we show the timing sequences for setting and resetting the module and in Figure 5.2 , we show the states of the flip-flop and the coupler switch during the set reset cycle. The state of the coupler switch is represented by the refractive index n as controlled by the output voltage of the flip flop Q. We note that it is not necessary to

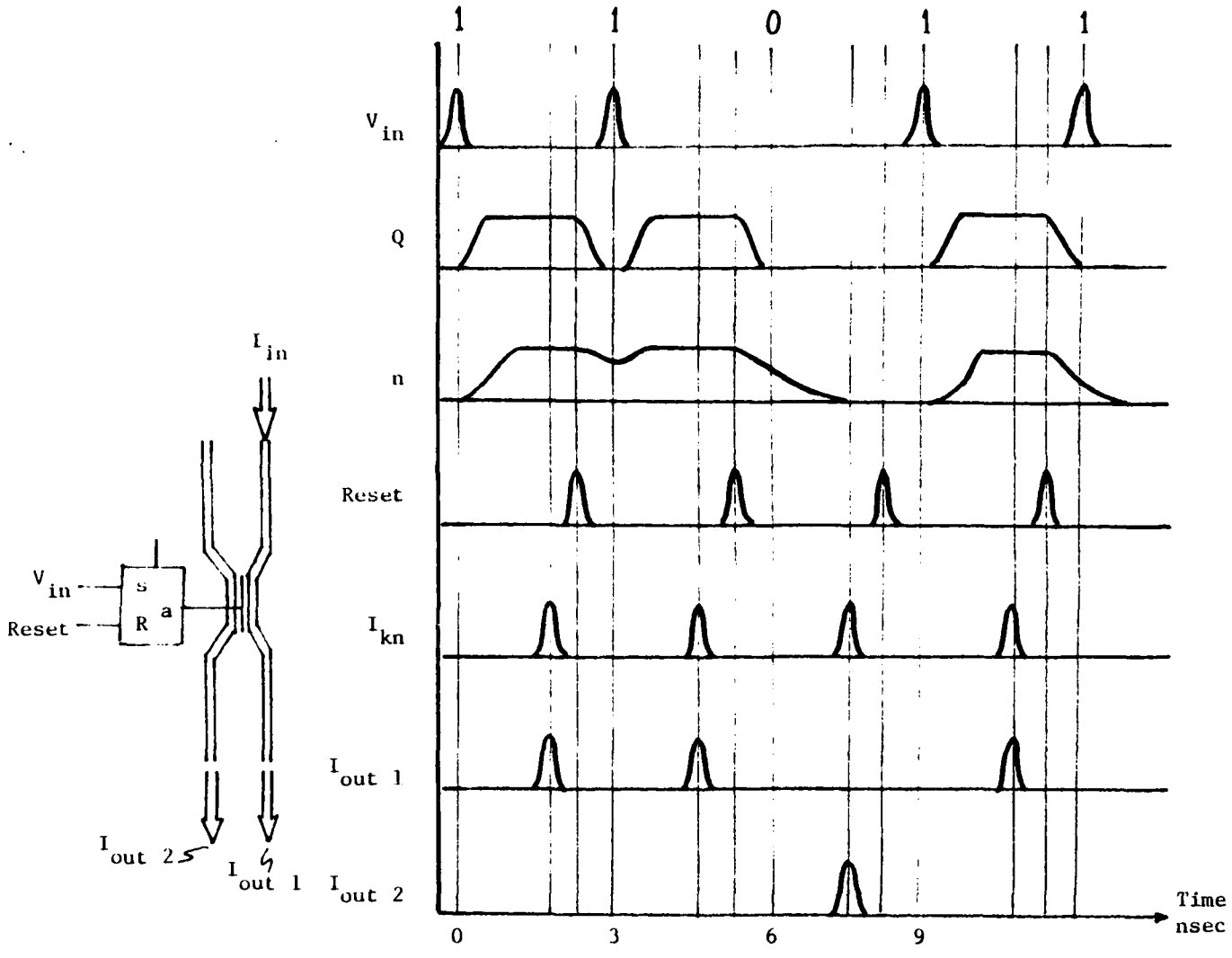


FIGURE 5.1 TIMING SEQUENCES FOR THE SETTING AND RESETTING OF MODULE.

A - 300 psec detect
B - 500 psec switch flip flop
C - 1 nsec switch coupler switch
D - 1.1 nsec "on" state
E - 500 psec reset flip flop
F - 100 psec gap between cycle
G - 1 nese reset coupler switch

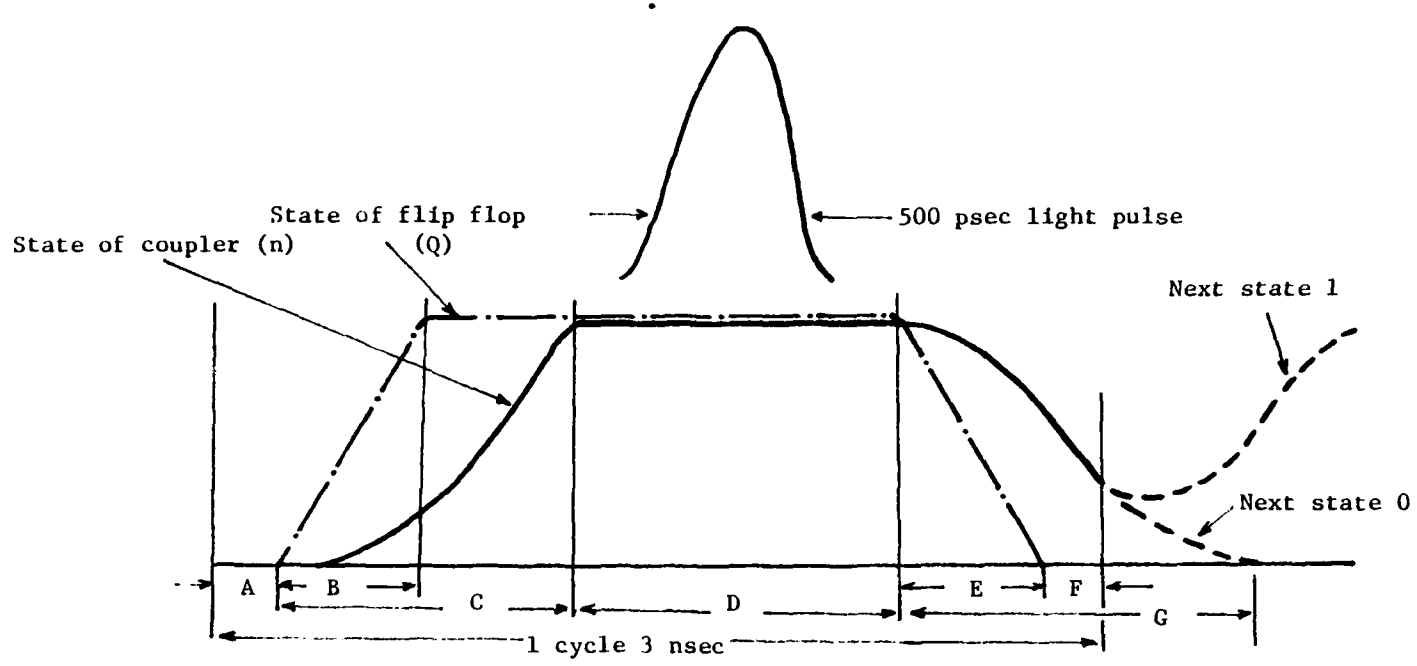


FIGURE 5.2 STATE OF COUPLER SWITCH DURING ONE CLOCK CYCLE.

provide a complete time period for the resetting of the coupler switch. As illustrated in Figure 5.2, if the next state is also '1', the refractive index of the coupler material will be driven up again and if the next state is '0', the flip flop output voltage Q will remain at zero and there would be ample time for the refractive index to be lowered before the light pulse is propagated through. The 100 psec gap between cycles is to provide a margin of safety in case the setting pulse of the next cycle occurs early and merges with the tail end of the resetting pulse, producing an ambiguous switching of the flip flop. This extra time period can be eliminated in the eventual development and refinement of the hardwares.

To compare these requirements with demonstrated hardware performances, we listed in Tables 5.1, 5.2, 5.3, and 5.4, the performance levels of some hardwares including light source, detectors and modulators that are possible candidates for the implementation of an optical numerical computer. And in Table 5.5, we listed the performances of the preferred hardwares for our design concept together with the performance requirements. We find the requirements and the performances of state of art hardwares are quite compatible. The major exceptions are the physical size and the optical loss through an optical switch.

The larger than desired physical size has 3 adverse effects. First, the propagation time of the light pulse through a module would be longer. Secondly, the optical loss through scattering and absorption is proportional to the length of the waveguide. A longer device would result in more optical power loss. Thirdly, a large device is simply undesirable if not unacceptable for many applications such as airborne processors.

Table 5.1 Performances of Bulk Modulators

	$Bi_{12}SiO_{20}$	DKDP	Liquid Crystal	Thermo-plastic	Photo-chromic	Silver Halide
Type of Address	Photo	Photo or Electron	Photo	Photo or Electron	Photo	Photo
Type of Modulation	Pockels Effect	Pockels Effect	Hybrid Field	Surface Deformation	Absorption	Absorption
Erase Mechanism	UV & Electric Field	UV & Electric Field	Electric Field	Heat	IR	N.A.
Cycle Time	5 msec	5 msec	40 msec	1 sec	1 sec	N.A.
Sensitivity	5 $\mu j/cm^2$	10 $\mu j/cm^2$	5 $\mu j/cm^2$	10 $\mu j/cm^2$	2 x 10^4 $\mu j/cm^2$	1 $\mu j/cm^2$
Storage Time	Short	Short	Short	Long	Short	Short
Life Time	> 10^7 Cycle	> 10^7 Cycle	10^4 Cycle	10^3 Cycle	10^5 Cycle	N.A.
Resolution	150-500 ℓ/mm	75 ℓ/mm	50 ℓ/mm	250 → 2000 ℓ/mm	2000	2000
Contrast Ratio	> 1000:1	100:1	100:1	100:1	1000:1	1000:1

Table 5.2 Performances of Integrated Modulators

	E.O. Modulators (KDP, ADP, CdTe)	$(PbM_0)_4$, Glass)	Directional Coupler ($LiNbO_3$)
Bandwidth (MHz)	2000	100	2000
Operation Voltage (V)	L.V. H.V. 300 3K	50	20
Power Dissipation (mw/MHz)	0.02	90	0.03
Extinction Ratio (dB)	33	30	25
Power Loss (-dB)	-0.2	-0.4	Alt $\Delta\beta$, Cobra -0.05, -1.2 plus -0.5 dB/cm

Table 5.3 Performances of Photodetectors

	Pin	APD
Peak Responsivity (A/W)	0.5	100
Sensitivity at 1 MHz (dBmW)	-60	-80
Noise Equivalent Power ($W/Hz^{1/2}$)	1×10^{-13}	1×10^{-15}
Rise Time (nsec)	1	0.2
Bandwidth (MHz)	1000	2000
Peak Response Wavelength (nm)	Si Ge 870 1400	Si Ge GaInAsP 880 1500 1000-17000
Quantum Efficiency (%)	85	50 → 85%
Gain	1	100
Bias Voltage (V)	50	300
Ave. Lifetime	10^6	10^6

Table 5.4 Performance of Light Sources

	LED		ILD	
Output (mw)	Large 7	Small 2	CW 20	Pulse 1000
Insertion Loss (-dB)	-18	-9	-3	
Modulation Bandwidth (MHz)	50	250	1500	0.1
Optical Bandwidth (nm)	35	22	2	2
Rise Time (nsec)	20	5	0.1	.05
Min. Drive Current (ma)	1	0.5	50	
Temperature Dependent	Moderate		High	
Ave. Lifetime	10^6		10^5	

Table 5.5 Performances of Preferred Hardwares

	Performance Required for 300 MHz Through-put Rate	Preferred Hardwares	Demonstrated Performance
Laser Pulse Width	500 psec	(ILD) Injection Laser Diode	1 nsec
Pulse Repetition Rate	300 MHz	Injection Laser Diode	300 MHz
Detect Response Time	200 psec	(APD) Avalanche Photodiode	100 psec
Optical Switch Switching Time	500 psec	Directional Waveguide Coupler	500 psec
Energy Dissip. per Switch (Modulation Power)	<10 μw/MHz	Directional Waveguide Coupler	100 μw/MHz
Optical Loss in Waveguide	~0.1 dB/cm	Out-Diffused $LiNbO_3$	0.5→1 dB/cm
Physical Size of Optical Switch	0.5 mm x 20 μm	Directional Waveguide Coupler	3 mm x 20 μm
Equivalent Propagation Time Through MOD 31 Module	50 psec	Directional Waveguide Coupler	290 psec
FLIP FLOP Switching Time	500 psec	Bipolar	500 psec

Other considerations: Average life time reliability component integration feasibility of mass production.

There are 2 basic causes of optical loss through an optical switch. As mentioned above, optical power is lost as it propagates through a waveguide, whether it is part of an optical switch or not. However, the number of optical switches would ultimately dictate the total length of waveguide in the computation module. Moreover, switching in general is not complete. Additional power is lost because not all the input light is switched to the desired channel. The power loss caused by such incomplete switching is quite high (-1.2 dB) for a simple coupler switch. The cross talk can be substantially decreased with the use of the alternate $\Delta\beta$ or the balanced bridge arrangements. However, the length of the device with these arrangements is also increased. This would result in a higher propagation loss that partially offsets the lower loss due to incomplete switching. If we assume a propagation loss of -0.7 dB/cm through a waveguide, the simple coupler has a minimum length of 3 mm and the total power loss would be (-1.2 dB) + (-0.3 x 0.7 dB) = -1.4 dB or 72% transmission. On the other hand, the balanced bridge arrangement has an incomplete switching loss of -0.04 dB and a minimum length of 9 mm. The total power loss though a switch would be (-0.04) + (-0.9 x 0.7 dB) = -0.67 dB or 86% transmission.

The pulse ILD can produce very narrow light pulses (< 100 psec) but the pulse repetition rate (< 100 KHz) is too low. The alternative is to pulse-modulate a cw ILD. Modulation frequency up to 1.5 GHz has been demonstrated for cw ILD. A chain of pulses can therefore be produced at 300 MHz with a pulse width of about 1 nsec, although it would be quite marginal for an optical computer system operating at 300 MHz clock rate.

Let us examine a more specific example, for an optical computer with 15 bits accuracy, moduli 2, 3, 5, 7, 11 and 13 can be used. Since 13 is the largest modulus, its implementation would determine the performance of the computer. Assuming that a sum of products operation is performed with the pipelined arrangements shown in Figure4.19, the light pulses have to propagate through a maximum of two computation modules each cycle. With either device, the light pulses passes through a maximum of m_i-1 optical switches and thus for modulus 13, the total would be 24 switches. If the simple couplers are used in the implementation, the transmission through each coupler switch is 72% and the minimum waveguide length is 3 mm. Propagating through 24 coupler switches, the total transmission would be 1% and the propagation delay would be 240 psec. If 1 mw of laser power is injected at the input, the optical output power would be 1 μw, well within the detectable level of an avalanche photodiode. The cycle time is equal to the set-reset time of a computation module plus the total propagation delay. Thus, the minimum cycle time for the computer would be 3.24 nsec, corresponding to a throughput rate of 300 MHz. If the optimized modules implemented with the minimum number of optical switches are used, the number of optical switches the light pulses have to be propagated through would be decreased to $\log_2 13 \cong 4$. The propagation delay is then reduced to 80 psec and a throughput rate up to 325 MHz would be possible.

5.3 COMPARATIVE ANALYSIS FOR SPECIAL PURPOSE ELECTRONICS AND OPTICAL NUMERICAL COMPUTERS

In a general purpose computer, it usually takes 4 cycles to complete an operation. To perform x + y for example, the instruction is first fetched from the instruction memory and the operation code is decoded. Next, the operand is readout from the data memory and entered into the arithmetic unit together with the operand from the

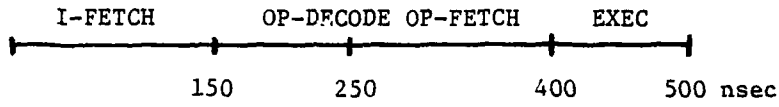


FIGURE 5.3 COMPLETE SET OF INSTRUCTION FOR AN ARITHMETIC OPERATION.

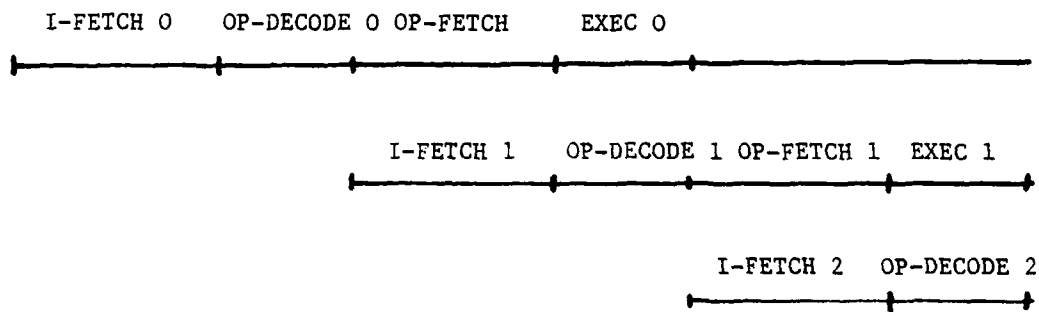


FIGURE 5.4 INCREASING THROUGHPUT RATE WITH PIPELINING.

input data register. In the last cycle, the arithmetic operation is performed and the result is put into the accumulator. Thus about 3/4 of time was spent in the noncomputation part of the operation. One approach to increase the throughput rate is to pipeline the process as illustrated in Figure 5.4. Such pipelining can only be fully utilized under two conditions. There must be separate memories for the instruction and coefficient data such that both can be fetched simultaneously. Secondly, there must be no logic decision (branch instruction) in the program. It is especially true if the branching is conditional, that is, the next instruction cannot begin until a decision is made on whether branching is to occur or not. Such a computer system would have less flexibility in programming but it is still very much programmable. The next step would be to eliminate the instruction fetch and operation code decode functions entirely. This can be done only if there is a fixed and limited set of instructions that have to be performed. What we now have is a class of highly specialized computers, capable of performing a very limited set of operations at high speed. These would include such systems as the FFT processors and various signal processing systems. In a way, they are very similar to their analog counterparts, performing only specific functions (e.g., low pass filtering). Flexibility has been sacrificed to gain speed. The speed can be further increased, at the expense of system complexity and cost, by using parallel structures. Hardwares are duplicated such that computations can be performed simultaneously in parallel instead of sequentially. The amount of parallelism (and therefore duplication) is determined by the throughput rate requirement. To have more parallism than necessary would be an inefficient use of hardware.

For most signal processing applications, the central operation that is performed repeatedly is

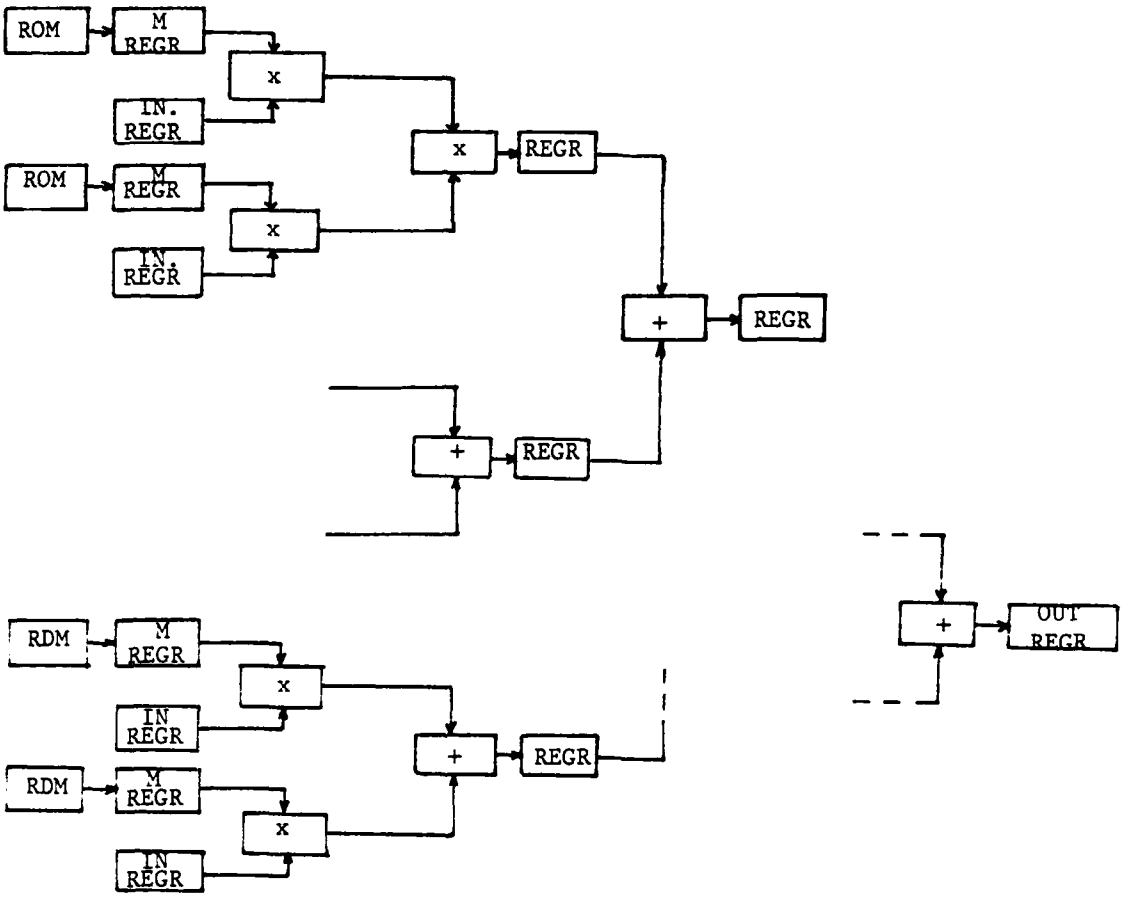


FIGURE 5.5 ELECTRONIC PARALLEL PROCESSOR.

$$\sum_{i=1}^{N} a_i(t)\ x_i(t)$$

Some important examples are correlation detection, FIR filtering and various transform operations such as DFT and Hadamard transform. To compare the performance of the two types of computer, it would therefore be realistic and instructive to use

$$\sum_{i=1}^{N} a_i(t)\ x_i(t)$$

as the computation goal.

To obtain a quantitative comparison, let us assume that N = 16 and determine the throughput rate for

$$f(t) = \sum_{i=1}^{16} a_i(t)\ x_i(t)$$

using the electronic and optical computers. For valid comparison, the electronic computer would be of a highly specialized type, capable only of performing a fixed set of computation and with a fully parallel architecture.

In Figure 5.5, the structure of the electronic computer is illustrated. It features parallel multipliers and data memories. The summations are performed with an adder tree. The operations are fully pipelined to obtain a $\frac{1}{T}$ throughput rate where T is the time required to perform the most time consuming instruction in the pipelined chain. The access time of the fastest ROM in the market is about 20 nsec, an 8 x 8-bit multiplication would require about 60 nsec and an 8 + 8-bit addition takes about 10 nsec. The throughput rate is therefore determined by the multiplication time. The initial delay in obtaining

the first output would be 6 x 60 = 360 nsec and the throughput rate would be $\frac{1}{60 \text{ nsec}}$ = 16.7 MHz.

The system speed can be further increased if we assume the following: 1) the coefficients a_i are fixed over the duration of the computation. This would be true for correlation and filtering operations; 2) the dynamic range M is sufficiently small that with a fixed set of a_i, the total number of possible results can be stored in N ROM memories as multiplication tables. Each of these ROM would have a capacity of M words. The multiplication operation is then performed through table look-up instead of using multipliers. The structure of such a system is illustrated in Figure 5.6 . The most time-consuming instruction is now the table look-up. Since the access time of a ROM is 20 nsec, the pipelined throughput rate would be 50 MHz and the initial delay is about 100 nsec.

The system we described above is highly specialized and it has very limited flexibility. However, it is optimum in terms of system speed. Any further improvement in speed will have to come through the use of faster hardwares.

To compare the performances with optical numerical computers, let us first examine the arrangement without pipelining as shown in Figure 4.17. For N = 16, the throughput rate would be

$$\frac{1}{3 + 17(.12) \text{ nsec}} = 198 \text{ MHz}$$

with an initial delay of 5 nsec. If the optimized computation modules are used, the throughput rate would then be

$$\frac{1}{3 + 17(.04) \text{ nsec}} = 272 \text{ MHz}$$

and the initial delay would be 3.68 nsec. We can see that the optical

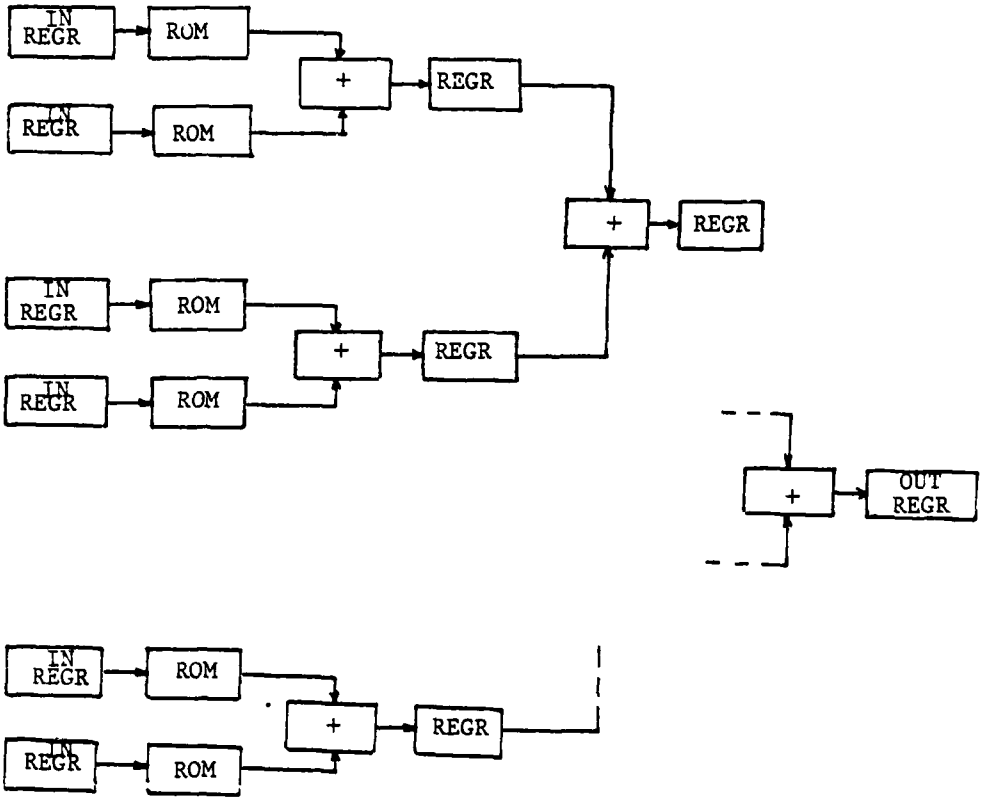


FIGURE 5.6 ELECTRONIC PROCESSOR USING TABLE LOOKUP.

residue computer enjoy a speed advantage both in terms of throughput rate and initial delay. The throughput rate can be further improved with the use of pipelining as shown in Figure 4.19. The throughput rate would be increased to

$$\frac{1}{3 + 2(.12)\ \text{nsec}} = 308\ \text{MHz}$$

with a 3.24 nsec initial delay using the basic multiplier and adders, and a throughput rate of

$$\frac{1}{3 + 2(.04)\text{nsec}} = 325\ \text{MHz}$$

with 3.08 nsec initial delay using the optimized versions.

We must be careful in interpreting any comparative studies between optical and electronic computers. The above discussion is valid only for the electronic hardwares chosen for comparison. Electronic computers can be constructed to produce much higher throughput rate using such new technologies as GaAs transfer electron devices. The use of these devices however, entails the use of much more complex systems. To compare optical and electronic system, we have to compare the respective curves of throughput rates as a function of complexities[25] as illustrated in Figure 5.7 . The integrated optics technology is too new to generate a reliable curve for the optical computer. However, it is likely that within a certain range, the optical computer can provide a higher throughput rate than its electronic counterpart for an equivelant level system complexity. We may also add that the system performance level estimated for the optical computer is valid only with the assumptions given for the hardware performances. The switching speeds of the flip-flop and optical switches are likely to improve with further development. The system throughput rate would also be improved accordingly.

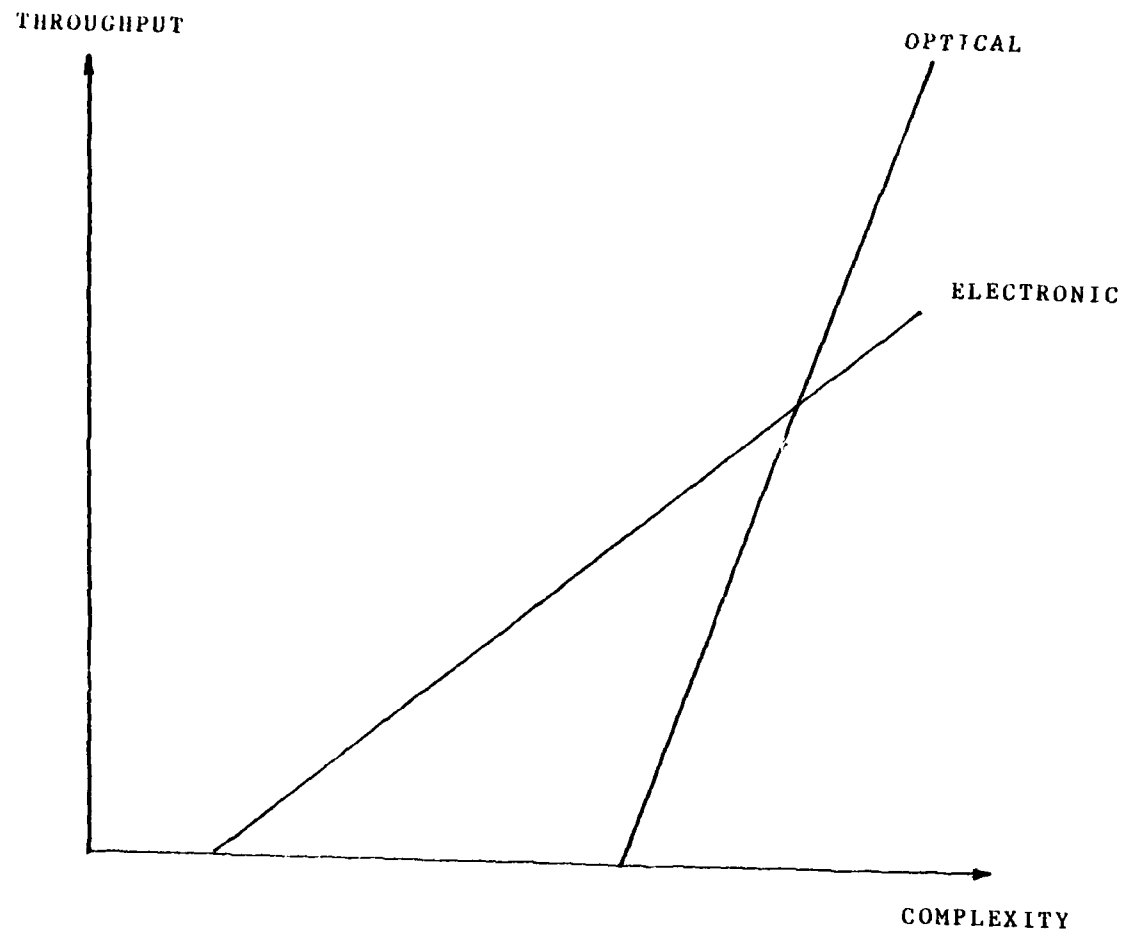


FIGURE 5.7 THROUGHPUT AS A FUNCTION OF COMPLEXITY FOR OPTICAL AND ELECTRONIC IMPLEMENTATIONS.

Beyond all the numbers, a point can be made that the optical residue computer is conceptually much simpler and in certain aspects it is fundamentally more efficient. A case in point is the multiplication by a number in storage. It eliminates the access time delay entirely and reduce the multiplication time by allowing the multiplication operation to be performed without carries and partial sums.

5.4 POSSIBLE SYSTEM APPLICATIONS FOR OPTICAL RESIDUE COMPUTER

The most attractive feature of the optical residue computer is obviously the high throughput rate and the relatively low level of complexity. Any practical application of the optical system should fully utilize its speed and parallel processing capability. Similar to its electronic counterparts, the applications would be of a specialized type. For example, the optical computer could be used as the front end processor for a radar system as shown in Figure 5.8. The function of the optical computer would be data reduction, performing such operations as FFT, filtering and correlation detection. The output data is decoded and entered into the central processing unit (CPU) where the data from various radars are correlated. The CPU would make the decisions and issue commands. Few or no decisions would be made by the optical computer and the program instructions and algorithms are fixed. However, the coefficients in the algorithm can be altered at any time by the CPU, changing for example, the filter functions.

Such as application would fit very well with the capabilities and limitations of the optical residue computer. It has the following features:

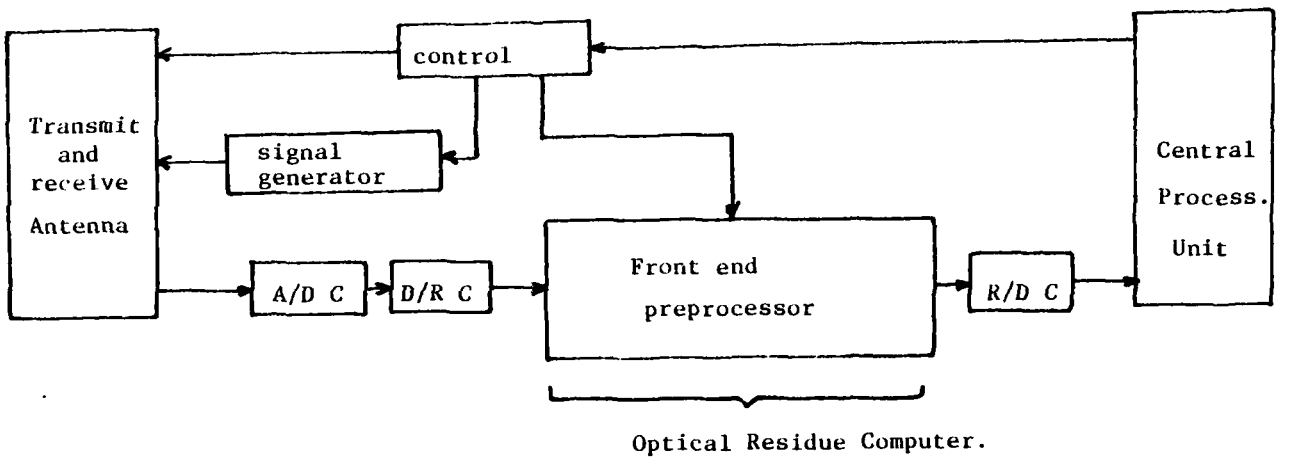


FIGURE 5.8 POSSIBLE SYSTEM APPLICATION FOR OPTICAL RESIDUE PROCESSOR.

(1) Program instructions are unchanged - the bulk of the computer time is often spent on addressing and decoding instructions. To eliminate these time delays, the program instructions must be fixed and limited.

(2) No branching instruction required in program - it is especially true if the branching is conditional. This condition is necessary in order to fully pipeline the optical computer.

(3) Constant and unidirectional data flow - these are also requirements for efficient pipelining of the system for real time processing.

(4) Computation algorithm suitable for parallel computation structure - one reason for the high computation speed of the optical residue computer is its parallel structure. Large number of operations are performed simultaneously. When using such a computer, the algorithm must be written in such a way that the parallism is taken full advantage of.

(5) No general division required - this requirement is unique to computers using the residue number system. Fortunately, many important computation algorithms can be written without any division operation.

6
DEVELOPMENTAL NEEDS

6.1 INTRODUCTION

In this section, we shall examine the development and advances in hardware technologies that are required before a numerical optical processor can become a practical reality. Some of these constitute only general improvements in the performance levels of present technology while others may require the development of better materials and fabrication techniques. However, none of these demand any scientific breakthrough.

While significant advances have been achieved in the last few years, the advances in integrated optics would have been even greater if there were better focus on the potential application of the integrated optics devices. This lack of clear direction for research has caused some companies and individual researchers to become hesitant in committing themselves in the field. The optical numerical computer may well be the needed impetus for the development of integrated optics, providing the research community with a viable application and specific area of research.

6.2 DEVELOPMENTAL NEEDS FOR IMPLEMENTATION OF MAPPING CONCEPT

In Section 4, we have presented a design for an optical processor based on the mapping concept. In the following we shall examine the immediate developmental needs to make the construction of such a processor practical, and the long term developmental needs to further improve the capabilities of the system.

6.2.1 IMMEDIATE DEVELOPMENTAL NEEDS

Most of the needed improvements are related to the physical size of the optical devices. Integrated optics today is at a comparable stage as the early transister in the field of electronics.

While the integrated optics devices are substantially smaller than their bulk counterparts, they have a long way to go before true miniaturizing and integration are achieved.

There are several advances in fabrication technology that must take place before the optical numerical computer concept can be realized:

(1) Integration of all necessary devices on the same chip. These would include laser diodes, optical switches, amplifiers, flip flops, and optical detectors. Such a development is quite feasible since all the devices can be constructed from the same basic substrate material, namely, GaAS.

(2) Development of fabrication technology to allow bending, overlapping, splitting and combining of waveguides with low optical loss and cross talk. This may require the development of new concepts, such as the etching of holographic gratings into corners of bent waveguide to allow abrupt deflection of guided light wave.

(3) The construction of identical optical switches. Since several devices have to be turned on by the same signal voltage, it would be necessary that the optical switches constructed be identical in their characteristics, especially the voltage required for complete switching.

With these developments in fabrication technology, an optical residue computer can be constructed. The first generation would likely have a throughput rate of about 100 MHz with a 15-bit accuracy (using for example moduli 2, 3, 5, 7, 11, 13). For a 100 MHz rate, the hardware performance requirements can be satisfied with a 500 psec detection time, a 3 nsec switching times for flip flops and coupler switches, and a 1 nsec laser pulse width. They are well within the demonstrated performance levels. To improve the throughput rate to 300 MHz for the second generation of optical

computer with 32 bit accuracy (using for example, moduli 2, 3, 5, 7, 11, 13, 17, 19, 23, and 29), the following improvements would be necessary:

(1) Reduction of optical loss through a waveguide switch to -0.5 dB/switch by decreasing the propagation loss in the waveguide and achieving more complete switching. For example, with a modulo 29 adder implemented without switch reduction, the light pulse has to propagate through a maximum of 28 switches per module. Thus, the transmission of optical power through each module would be about 4%. Using the pipeline concept, the light pulse has to propagate through only 2 modules before its detected. If for example, 0.1 mw of the optical power is coupled into the waveguide of the first module, the light pulses detected at the output of the second module would have an optical power of 158 nw, well above the detectable level of an avalanche photodiode.

(2) Reduction of laser pulse width. To maintain a 300 MHz throughput rate, a pulse width under 0.5 nsec for the laser pulse would be desirable. This would require improvements for both the laser diode and the electronic driving circuits.

6.2.2 LONG TERM DEVELOPMENT NEEDS

The first and second generation computers can be built upon the concept we presented in this report. For the third generation computer, more radical technology development might be needed. We list in the following, some of the possible developments that could substantially improve the performance level of the optical numerical computer:

(1) Reduction of the physical size and optical loss of optical switches with the development of new electro-optical materials. The coupling length of a coupler switch and the amount of cross talk are ultimately determined by the electro-optical material. To further decrease the physical size, cross talk and propagation delay, new electro-optical materials must be developed.

(2) The production of mode-locked laser diodes for the generation of very narrow pulses (< 100 psec) at high repetition rate (>1 GHz). Such a laser diode would make possible the development of an optical computer with a throughput rate over 1 GHz.

(3) Reduction or elimination of electronic devices and optical-electronic interfaces with the use of fast optically activated switches. Most of the fast optical switch existing today are electrically activated. The use of photo detectors for optical-electrical conversion and electronic devices such as amplifiers and flip flops is therefore necessary for the control of the switches. Such hybrid approach is not the most efficient in terms of speed. However, it is and likely to be for some time, the most practical and realizable approach. One reason is the difficulty in producing a fast optically activated device. First of all, the switching speed is related to the optical power activating the material and the amount of light power, especially in integrated optics, is very limited. Secondly, optical intensity cannot be amplified as easily as electrical voltage. A scheme utilizing stimulated emission may be necessary to provide amplification and high speed switching capability.

(4) Wavelength multiplexing - this would allow two or more computations to be carried out simultaneously with the same hardwares. Alternatively, the same computation but for different moduli (e.g., modulus 13 and modulus 11) can be performed together in the same computation module. This would further increase the packing density of the optical computing system.

6.3 DEVELOPMENTAL NEEDS FOR IMPLEMENTATION OF CYCLIC CONCEPT

Most cyclic devices such as phase shifter, modulators are analog devices. To utilize them for numerical operations would require the quantization of the control signals. However, quantization always involves certain probability error. In sequential operations, incremental errors will accumulate and the probability of error can easily be built up to an unacceptable level. To avoid such accumulation of errors, a device that would automatically adjust itself to the desired quantized state would be necessary. One approach is to develop a cyclic device that exhibits multistable state behavior. Several electro-optics devices have been proposed using feedback arrangements that would produce multistable states behavior. However, with the use of feedback the inherent cyclic characteristic is generally lost. The ideal characteristic of a cyclic device for residue arithmetic is illustrated in Figure 6.1. A device with such a characteristic still awaits development.

Even with an ideal cyclic device, the only operation that can be performed directly would be addition and subtraction. Multiplication can be performed by successive addition but such an approach is not efficient or practical if the modulus is large. It is also difficult to perform transformations and other common functions such as x^n, $\frac{1}{x}$, and -x which can be implemented easily by the mapping approach using fixed maps. One solution is to combine the cyclic and mapping concepts. utilizing mapping devices in performing transformations and specifications and the cyclic devices for the addition and subtraction operations.

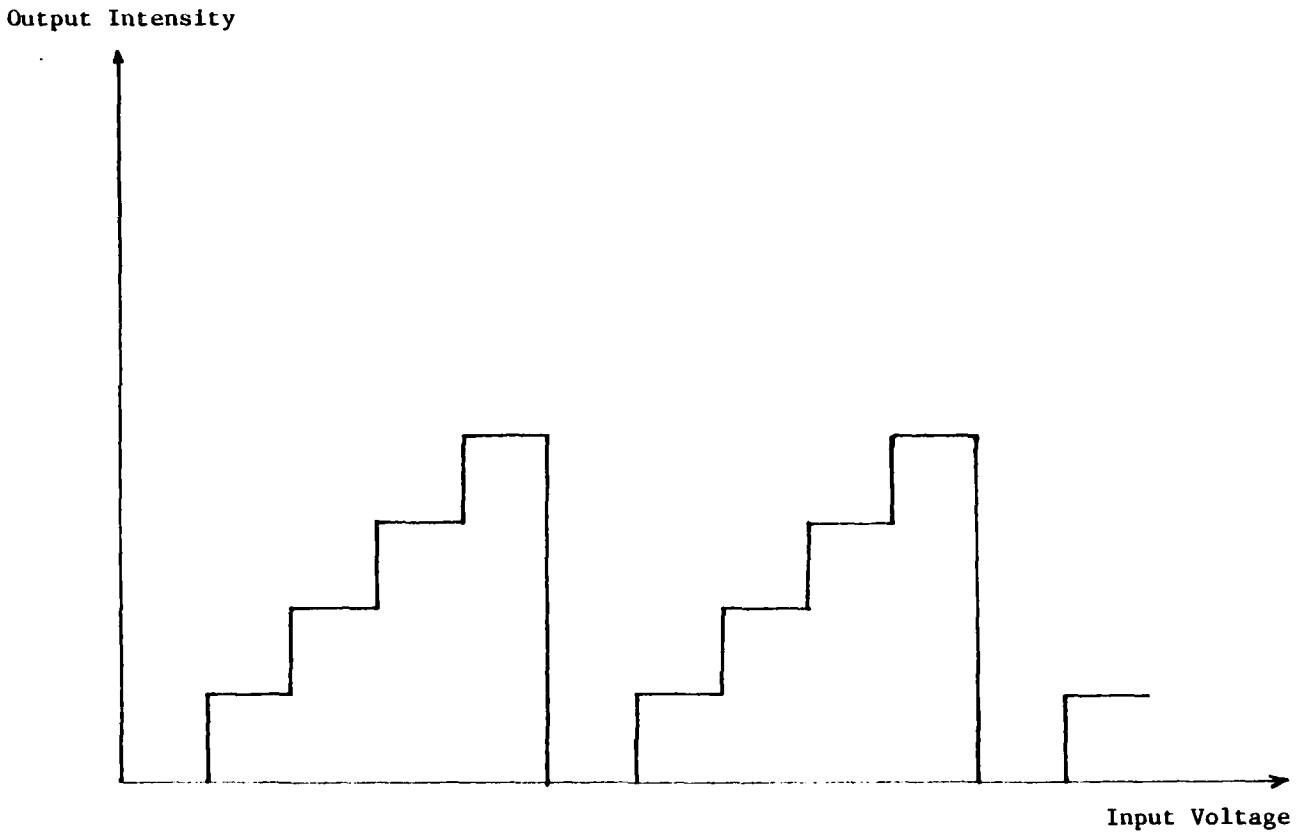


FIGURE 6.1 IDEAL CHARACTERISTIC OF CYCLIC DEVICE FOR MODULUS 5.

7
DEVELOPMENTAL PLAN

7.1 INTRODUCTION

The results of our study indicate that the concept of a numerical optical processor can be a viable alternative to conventional electronic approaches in improving the computation speed and possibly packing density and power consumption without a substantial increase in system complexity. Although the optimum implementation approach for a numerical optical processor cannot be defined concisely at this time due to the early stage of our investigation, it is clear that the mapping (or table look up) and cyclic approaches are the two most promising directions available. The eventual best approach will be influenced strongly by the development of key components directed to the numerical optical processor. From these points of view, we suggest as a first phase of development program three tasks. These tasks deal with the refinement of the system design concept, key components development and the fabrication of a basic functional computing unit which would serve as the foundation of the total system development in the second phase.

7.2 SYSTEM DESIGN CONCEPT

The objective of this task is to refine the present optical numerical processor design and to produce a system design for a specific application. This would include the design development of input output interfaces, data buses, programming controls and data storages. This effort would be directed to both the mapping and cyclic implementation approaches. In addition, new design approaches will also be explored, including system that are not based on the residue number system. One specific approach that may be looked into is the use of electro optical devices to implement large array of binary logic gates, particularly NOR gates.

For the mapping implementation of a residue based optical processor, the development efforts would include the following:

1) Establishment of the optimum mapping structure and the definition of a systematic design method for the computation modules. The design will be optimized for speed and physical size.

2) Streamlining of processor architecture for maximum throughput rate.

3) Updating the selection of hardware components and the initiation of contacts and consultations with leading researchers and manufacturers.

4) Selection of a specific processor application and the production of a comprehensive system design. System performances will be evaluated through computer simulation.

For the cyclic approach, efforts under this task would include the following:

1) Establishment of a choice of materials and feedback techniques for the generation of multistable states behavior for quantized operations.

2) Development of efficient methods for performing multiplication, division and table look up with cyclic devices.

3) Development of design concepts for the efficient and high speed coupling between cyclic and mapping devices.

4) Selection of a specific application and system design based on the cyclic approach. A combination of cyclic and mapping devices may also be used.

For the non residue approaches, the possibility of implementing a large array of binary logic gates with electro optic devices will

be investigated. This is motivated by the potential of high packing density and system throughput with its two-dimensional parallel processing capability. This work will include the following:

1) Reviewing current electronic digital techniques for parallel processing.

2) Reviewing proposed optical techniques for logic gates implementation.

3) Establishment of the most suitable architecture for optical implementation.

4) Development of a design for large optical logic arrays and estimates of performance characteristics.

5) Selection of a specific processor application and establishing a system design configuration using optical logic devices.

7.3 COMPONENT DEVELOPMENT

Concurrent with the development of the system concept, an intensive developmental program for key components will be carried out. Most of these components have been demonstrated and required only further refinement to improve the performance levels. Others may require the development of new materials, fabrication techniques and engineering approaches.

For the mapping implementation, the following development program will be performed:

1) Refinement of existing electro optic switches. The goal is to decrease optical loss, cross talk, physical size and operating voltage.

2) Development of fabrication technology to allow overlapping, splitting, splicing and bending of waveguides with low loss.

3) Development of integrated mode-locked laser diode.

4) Refinement of avalanche photodiode including the improvement of sensitivity, signal to noise ratio, gain, response time and the lowering of bias voltage.

5) Development of high density optical ROM for fixed map transformation and storage of reference coefficients. This may include the use of holographic optical memories.

6) Development of new electro optic material to decrease the physical size and improve the switching performance of optical switches.

7) Integration of all key components using GaAs as the basic substrate material.

For the cyclic implementation, the following developmental program will be undertaken:

1) Refinement of existing feedback technique to produce multistable states behavior. The efforts will be geared towards the increase in the number of stable states and the reduction of hysteresis effect.

2) Development of a cyclic device with multistable states behavior. This may be achieved with the use of electronic comparator and triggering circuits together with the feed back techniques above.

3) Development of new electro optic material to improve the switching speed and dynamic range.

Beside the refinement of components for the arithmetic computation units, key components for other functional units will also be developed. This will include the input interfaces, storage devices, timing and programming controls.

7.4 DEMONSTRATION UNIT DEVELOPMENT

According to the results of the component development task, the system and component designs will be finalized. The designs will be based on available hardwares and fabrication technologies. A small demonstration unit will then be constructed with the aid of subcontractors. The word length will be limited to 8 bits and the computation will be of a fixed and nonrecursive type. The unit is not intended to be used in an actual operating system but as a demonstration unit for the evaluation of the designs and hardware implementations of the computing units, interfaces and controls. From the results of these evaluations, the system and component designs will be modified and improved. A plan for the construction of a prototype numerical optical computing system will then be produced for a specific BMD application.

REFERENCES

1. A. Huang, Y. Tsunoda, J.W. Goodman and S. Ishihara, "Optical Computation Using Residue Arithmetic," Applied Optics, 18, p. 149, 1979.
2. N.S. Szabo and R.I. Tanaka, Residue Arithmetic and Its Application to Computer Technology, McGraw Hill, New York, 1967.
3. P.W. Gheney, "A Digital Correlator Based on the Reside Number System," IRE Trans. Electron Comp., EC-10, P. 63, 1961.
4. D.K. Banerji, "On the Use of Residue Arithmetic for Computation," IEEE Trans. Comp., 23, p. 1315, 1974.
5. Y.A. Keir, P.W. Cheney, and M. Tannenbaum, "Division and Overflow Detection in Reside Number Systems," IRE Trans. Electron. Comp., EC-11, p. 501, 1962.
6. H.L. Garner, "The Residue Number System," IRE Trans. Electron Comp. EC-8, p. 140, 1959.
7. M. Valach, "Origin of the Code and Number System of Residue Classes," Technical Information Center, MCLTD, Wright-Patterson AFB, Ohio, F-TS-10126/V, 1955.
8. S.A. Collins, "Numerical Optical Data Processor," Proc. Soc. Photo. Opt. Instr. Eng., 128, p. 149, 1979.
9. M. Okada and K. Takizawa, "Multistable States Device with Optical Feedback," IEEE J. Quantum Elec., QE-15, p. 82, 1979.
10. D. Psaltis and D. Casasent, "Correlation Approach to Optical Residue Computing," Applied Optics, 18, p. 163, 1979.
11. A. Huang, "The Implementation of a Residue Arithmetic Unit Via Optical and Other Physical Phenomena," Proc. Int. Opt. Comp. Conf., Washington, D.C., p. 14, 1975.
12. I. Kaminow, "Optical Waveguide Modulators," IEEE Trans. Micro. Theory and Techniques, MTT-23, p. 57, 1975.
13. B.J. Chang, "Grating Based Interferometers and Their Applications," ERIM Report 671034-2-X, 1974.
14. C.S. Tsai, L.T. Nguyen and C.C. Lee, "Electro Optic Phase-Array Light Beam Deflector with Application to Analog-to-Digital Conversion," Digest of Technical Papers, Topical Meeting on Integrated and Guided Wave Optics, Salt Lake City, TUC2-1, 1977.
15. S.K. Cheem, "Total Internal Reflection Integrated-Optics Switch; a Theoretical Evaluation," Applied Optics, 17, p. 3679, 1978.

16. H. Taylor and A. Yariv, "Guided Wave Optics," Proc. of IEEE, 62, p. 1044, 1974.

17. A Carenco, et. al., "Directional Coupler Switch in Moleculr-Beam Epitaxy GaAs," Applied Physics Letters, 34, p. 755, 1979.

18. P.S. Cross, R.V. Schmidt and R.L. Thornton, "Optically-Controlled Two Channel Integrated-Optical Switch," Digest of Technical Papers, Topical Meeting on Integrated and Guided Wave Optics, Salt Lake City, TuB1-1, 1978.

19. V. Ramaswamy and M. Divino, "A Balanced Bridge Modulator Switch," Digest of Technical Papers, Topical Melting on Integrated and Guided Wave Optics, Salt Lake City, TuA4-1, 1978.

20. H. Kogelnik and R.V. Schmidt, "Switched Directional Couplers with Alternating $\Delta\beta$," IEEE J. of Quan. Electronics, QE-12, p. 396, 1976.

21. A. Tai, I. Cindrich, J.R. Fienup and C.C. Aleksoff, "Optical Residue Computer with Programmable Computation Modules," to be published in Applied Optics, 1979.

22. I. Cindrich, A. Tai, J.R. Fienup and C.C. Aleksoff, "Numerical Optical Processor Concepts," to be published in the Proceedings of the Soc. Photo Opt. Instr. Eng., 1979.

23. H. Kogelnik, "An Introduction to Integrated Optics," IEEE Trans. on Micro Theory and Technologies, MTT-23, p. 2, 1975.

24. H. Taylor, "Guided Wave Electro Optic Devices for Logic and Computation," Applied Optics, p. 1493, 1978.

25. A. Huang, J. Mandeville, J.W. Goodman and S. Ishihara, "System Considerations in the Design of Residue Processors," Stanford Electronics Laboratories Technical Report L722-2, 1979.